Al-Azhar University

Faculty of Engineering

Electrical Engineering Department

(Electronics & Communication)

B.Sc.

GRADUATION PROJECT

Building Ad Hoc Networks Using Wireless Sensor Networks

Supervised by :

Prof. Abdellatif I. Moustafa

PRESENTED BY:

Mohamed Saeed Elsayed (T.L)

Ahmed Elsayed Ramadan Abdelhamid

Mohammed Taha Ahmed Aboelhoda

Anas Moustafa Abdelhamid Ghazy

Mohamed Abdelfatth Mohamed Behiry

Islam Ahmed Elsafy Ahmed

July 2026

Preface

The continuous growth of renewable energy systems, particularly solar Photovoltaic/Thermal (PV/T) installations, has created a pressing need for intelligent, low-cost, and reliable monitoring solutions capable of tracking the electrical and thermal behaviour of these systems in real time. Solar panels are highly sensitive to environmental conditions, and their performance can degrade significantly without an operator ever being aware that a problem has occurred. This graduation project, **Building an Ad hoc Network using Wireless Sensor Networks** , was undertaken to address this need by designing and implementing a wireless monitoring system capable of continuously observing the key electrical and thermal parameters of a solar PV/T installation and reporting them to the user in real time.

The work presented in this report documents the complete journey of the project, from the theoretical foundations of computer networks and wireless sensor networks, through the formal formulation of the monitoring problem, to the practical design, configuration, and implementation of a fully functional WSN-based monitoring system. The implemented system is built around a National Instruments (NI) WSN platform, consisting of an **NI WSN-9791 Gateway** and two wireless sensor nodes, an **NI WSN-3212** node and an **NI WSN-3202** node, which together acquire temperature, insolation, and the voltage and current of both the solar array and the load. The acquired data is processed and visualised using **LabVIEW**, and critical readings are relayed to the user through a Telegram-based alert channel, enabling remote, real-time awareness of the system's status from any location.

This report is organised into five chapters. **Chapter 1** introduces the fundamental concepts of computer networks, including network components, topologies, and the different types of networks that form the broader context in which wireless sensor networks operate. **Chapter 2** narrows the focus to Wireless Sensor Networks specifically, covering their definition, architecture, protocol stack, and the advantages and challenges associated with their design. **Chapter 3** introduces LabVIEW as the software platform chosen for data acquisition and processing, explaining its role in interfacing with the WSN hardware and presenting the acquired data to the user. **Chapter 4** presents the formal problem formulation that guided the design of the implemented system, translating the practical requirements of monitoring a solar PV/T installation

into a rigorous mathematical model, complete with objective functions, design constraints, and a numerical case study based on the actual hardware used. Finally, **Chapter 5** documents the complete design and implementation of the system, covering the WSN node architecture, the network topology, the hardware configuration performed through NI Measurement & Automation Explorer (NI MAX), the LabVIEW software implementation, and the Telegram monitoring channel used to deliver live alerts.

It is hoped that this report not only documents the technical work carried out during this project, but also serves as a clear and useful reference for anyone seeking to understand how wireless sensor networks can be practically applied to the monitoring of renewable energy systems.

Contents

List of Tables **17**

1.1 Network Definition....22

1.2 Network Components....23

1.3 Computer Network Types....24

1.3.1 Local Area Networks (LANs)....24

1.3.2 Metropolitan Area Networks (MANs)....25

1.3.3 Wide Area Networks (WANs)....26

1.4 Network Topology....27

1.4.1 Point-to-Point Topology....27

1.4.2 Bus Topology....28

1.4.3 Ring Topology....29

1.4.4 Mesh Topology....30

1.4.5 Star Topology....31

1.5 Network Model....32

1.5.1 Open Systems Interconnection (OSI) Model....32

1.5.2 TCP/IP Model....39

1.6 Network Devices....40

1.6.1 Hub....40

1.6.2 Repeater....41

1.6.3 Bridge....42

1.6.4 Switch....43

1.6.5 Router....44

1.6.6 Modem....44

1.6.7 Network Interface Card (NIC)....45

1.6.8 Firewall 46
1.6.9 Wireless Access Point (WAP) 47
1.6.10 Server 47
1.7 Communication Systems 49
1.7.1 Wired versus Wireless Communication 50
1.7.2 Wireless Sensor Networks (WSN) 52
1.7.3 Ad Hoc Networks 54
2.1 Definition of a Wireless Sensor Network 58
2.2 Importance and Applications of Wireless Sensor Networks 61
2.2.1 Importance of Wireless Sensor Networks 61
2.2.2 Applications of Wireless Sensor Networks 62
2.3 Basic Components of a Wireless Sensor Network 68
2.3.1 Sensor Node 68
2.3.2 Base Station (Sink Node) 71
2.3.3 Communication Network 72
2.3.4 Task Manager (End-User Application) 72
2.4 WSN Network Architecture 73
2.4.1 Flat (Single-Tier) Architecture 73
2.4.2 Hierarchical (Clustered) Architecture 74
2.4.3 Hybrid Architecture 75
2.5 Data Flow in WSN 76
2.5.1 Sensing 77
2.5.2 Local Processing 77
2.5.3 Communication and Routing 77
2.5.4 Data Aggregation and Delivery 78
2.6 WSN Protocol Stack 78
2.6.1 Physical Layer 79
2.6.2 Data Link Layer 79
2.6.3 Network Layer 79
2.6.4 Transport Layer 79
2.6.5 Application Layer 80
2.6.6 Cross-Layer Management Planes 80

2.7 Design Considerations of WSN Architecture..... 80
2.8 Advantages of WSN Architecture.....82
2.9 Challenges of WSN Architecture.....83
Conclusion..... 84
3.1 Overview of LabVIEW.....86
3.1.1 History and Development..... 86
3.1.2 LabVIEW Programming Model..... 87
3.2 LabVIEW Key Components and Features.....88
3.2.1 Virtual Instruments (VIs)..... 88
3.2.2 Data Types and Wires..... 88
3.2.3 Structures..... 88
3.2.4 DAQ (Data Acquisition) Subsystem..... 89
3.3 LabVIEW in Wireless Sensor Networks..... 90
3.3.1 Role of LabVIEW in WSN Applications.....90
3.3.2 Communication Interfaces..... 91
3.4 Data Acquisition and Processing in LabVIEW..... 93
3.4.1 Data Acquisition Pipeline..... 93
3.4.2 Signal Processing.....94
3.4.3 Data Logging and File I/O..... 95
3.5 LabVIEW Front Panel Design for WSN Monitoring.....95
3.5.1 User Interface Components.....95
3.5.2 Alarm and Threshold Management.....96
3.6 Integration of LabVIEW with the WSN System..... 97
3.6.1 System Architecture.....97
3.6.2 Data Flow in the Integrated System..... 98
3.6.3 LabVIEW VI Structure for This Project..... 99
3.7 Advantages of Using LabVIEW in WSN Projects.....99
4.1 Motivation for a Formal Problem Statement.....102
4.2 Hardware Reference Specifications..... 103
4.3 System Model and Deployment.....104
4.3.1 Network Topology Representation.....104
4.3.2 Spatial Deployment Model..... 104

4.3.3 Adjacency and Connectivity Representation....106
4.3.4 Node Degree and Gateway Load....106
4.3.5 Rationale for Cluster-Based Spatial Partitioning....107
4.4 Objective Functions....108
4.4.1 Objective 1 — Maximisation of Network Lifetime....108
4.4.2 Objective 2 — Minimisation of Energy Consumption....109
4.4.3 Objective 3 — Maximisation of Data Reliability....111
4.5 Design Constraints....113
4.5.1 Energy Constraints....113
4.5.2 Communication and Range Constraints....114
4.5.3 Sampling and Data-Rate Constraints....116
4.6 Decision Variables....118
4.6.1 Mapping of Decision Variables to the LabVIEW Implementation....118
4.7 Numerical Case Study....120
4.7.1 Deployment Configuration....120
4.7.2 Aggregate Data-Rate Verification....120
4.7.3 Energy Budget Verification....121
4.8 Sensitivity Analysis and Design Trade-offs....122
4.8.1 Energy Lifetime versus Sampling Resolution....122
4.8.2 Network Lifetime versus Reliability....122
4.8.3 Alert Responsiveness versus Channel Utilisation....123
4.8.4 Summary of Trade-off Resolutions....123
4.9 Notation Summary....125
5.1 LabVIEW Project Creation and Target Configuration....127
5.1.1 Project Creation....127
5.1.2 Adding the WSN Gateway and Node....128
5.1.3 Deploying the Gateway and Node....130
5.1.4 Verifying the Node in Measurement & Automation Explorer....131
5.2 Monitoring Virtual Instrument (VI) Development....132
5.2.1 Block Diagram....132
5.2.2 Front Panel....132
5.3 Experimental Range and Link Quality Testing....133

5.3.1 Test Case 1: Short Range (20 cm)..... 134
5.3.2 Test Case 2: Distance Greater Than 20 m..... 135
5.3.3 Test Case 3: Distance Greater Than 40 m, With Walls..... 136
5.3.4 Test Case 4: Distance Greater Than 50 m..... 137
5.3.5 Summary of Range Test Results..... 137
5.4 Multi-Node System: Thermocouple and Soil Moisture Sensors..... 138
5.4.1 Sensor Assignment..... 139
5.4.2 Updated Block Diagram..... 139
5.4.3 Updated Front Panel..... 139
5.5 Host Controller VI and High-Temperature Alert System..... 141
5.5.1 Block Diagram and Alarm Logic..... 141
5.5.2 Front Panel Dashboard..... 142
5.5.3 Telegram Alert Script..... 145
5.5.4 Alert Verification..... 146
5.6 Conclusion..... 146

List of Abbreviations

Abbreviation	Definition
ACK	Acknowledgement
ADC	Analog-to-Digital Converter
AP	Access Point
API	Application Programming Interface
ARPANET	Advanced Research Projects Agency Network
ASCII	American Standard Code for Information Interchange
AWS	Amazon Web Services
BGP	Border Gateway Protocol
BLE	Bluetooth Low Energy
BS	Base Station
CJC	Cold-Junction Compensation
CRC	Cyclic Redundancy Check
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance
CSV	Comma-Separated Values
DAQ	Data Acquisition
DARPA	Defense Advanced Research Projects Agency
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name System
DSL	Digital Subscriber Line
EIGRP	Enhanced Interior Gateway Routing Protocol
FDDI	Fiber Distributed Data Interface
FND	First Node Death
FTP	File Transfer Protocol
GPS	Global Positioning System
GUI	Graphical User Interface
GW	Gateway
HMI	Human-Machine Interface

Abbreviation	Definition
HND	Half Node Death
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IB	Battery Current
ID	Identifier
IEEE	Institute of Electrical and Electronics Engineers
IL	Load Current
IMAP	Internet Message Access Protocol
Insol	Insolation (Solar Irradiance)
IoT	Internet of Things
IP	Internet Protocol
ISA	Array Current
ISM	Industrial, Scientific, and Medical (radio band)
ISP	Internet Service Provider
LAN	Local Area Network
LED	Light-Emitting Diode
LND	Last Node Death
LTE	Long-Term Evolution
MAC	Media Access Control
MAN	Metropolitan Area Network
MANET	Mobile Ad hoc Network
MCU	Microcontroller Unit
MPLS	Multiprotocol Label Switching
NAT	Network Address Translation
NI	National Instruments
NI MAX	NI Measurement & Automation Explorer
NIC	Network Interface Card
ODBC	Open Database Connectivity
OSI	Open Systems Interconnection (model)
OSPF	Open Shortest Path First
PDR	Packet Delivery Ratio

Abbreviation	Definition
PL	Path Loss
PV	Photovoltaic
PV/T	Photovoltaic/Thermal
QoS	Quality of Service
RAID	Redundant Array of Independent Disks
RAM	Random Access Memory
RF	Radio Frequency
RSSI	Received Signal Strength Indicator
SCADA	Supervisory Control and Data Acquisition
SMTP	Simple Mail Transfer Protocol
SNMP	Simple Network Management Protocol
SOC	State of Charge
SQL	Structured Query Language
SRAM	Static Random Access Memory
SSID	Service Set Identifier
STP	Spanning Tree Protocol
SYN	Synchronize (TCP control flag)
TCP	Transmission Control Protocol
TCP/IP	Transmission Control Protocol / Internet Protocol
TDMA	Time Division Multiple Access
TDMS	Technical Data Management Streaming
TLS/SSL	Transport Layer Security / Secure Sockets Layer
UART	Universal Asynchronous Receiver-Transmitter
UDP	User Datagram Protocol
USB	Universal Serial Bus
UTP	Unshielded Twisted Pair
VANET	Vehicular Ad hoc Network
VI	Virtual Instrument (LabVIEW)
VISA	Virtual Instrument Software Architecture
VL	Load Voltage
VPN	Virtual Private Network

Abbreviation	Definition
VSA	Array Voltage
WAN	Wide Area Network
WAP	Wireless Access Point
WSN	Wireless Sensor Network

List of Figures

1.1 A typical computer network showing Internet, Firewall, Wireless Router, and connected end devices..... 22

1.2 Local Area Network (LAN) — multiple devices connected through a router to the Internet.....25

1.3 Metropolitan Area Network (MAN) — multiple LANs (Hospital, College, Factory, School) interconnected through a central MAN..... 26

1.4 Wide Area Network (WAN) — multiple LANs interconnected across geographically distant locations through dedicated server links..... 27

1.5 Point-to-Point Topology — a dedicated link connecting two devices directly..... 28

1.6 Bus Topology — all devices connected to a single shared backbone cable.....29

1.7 Ring Topology — devices connected in a closed circular loop..... 30

1.8 Mesh Topology — every device connected to every other device through dedicated links.....31

1.9 Star Topology — all devices connected to a central switch.....32

1.10 The Seven Layers of the OSI Model..... 33

1.11 OSI Layer 7 — Application Layer protocols and services..... 34

1.12 OSI Layer 6 — Presentation Layer: encryption, compression, and translation.....34

1.13 OSI Layer 5 — Session Layer: establishing and managing communication sessions.....35

1.14 OSI Layer 4 — Transport Layer: TCP vs UDP and segmentation process..... 36

1.15 OSI Layer 3 — Network Layer: IP addressing and packet routing..... 37

1.16 OSI Layer 2 — Data Link Layer: MAC addressing and frame structure..... 38

1.17 OSI Layer 1 — Physical Layer: bit transmission over physical media..... 39

1.18 TCP/IP Model layers compared to the OSI Model.....40

1.19 Network Hub — broadcasts data to all connected devices..... 41
1.20 Repeater — regenerates and amplifies signals to extend network coverage.....42
1.21 Bridge — connects two network segments and filters traffic using MAC addresses. 42
1.22 Network Switch — intelligently forwards data to the intended recipient using MAC addresses.....43
1.23 Router — forwards packets between networks and connects LANs to the internet. 44
1.24 Modem — bridges a local network to the internet via the ISP infrastructure..... 45
1.25 Network Interface Card (NIC) — connects a device to the network..... 45
1.26 Firewall — monitors and filters network traffic to protect the internal network.....46
1.27 Wireless Access Point (WAP) — provides Wi-Fi connectivity to wireless devices. 47
1.28 Server — provides centralised resources and services to client devices over the network.....48
1.29 Basic Communication System — transmitter, channel, and receiver..... 49
1.30 Wired vs Wireless Communication — comparison of the two transmission approaches.....51
1.31 Wireless Sensor Network (WSN) — distributed sensor nodes collecting data and relaying it to a base station.....54
1.32 Ad Hoc Network — nodes communicating directly without any fixed infrastructure..... 56
2.33 General overview of a Wireless Sensor Network (WSN) — sensor nodes deployed across an area, communicating wirelessly with a base station..... 58
2.34 Application domains of Wireless Sensor Networks — from environmental monitoring and precision agriculture to healthcare and smart cities..... 60
2.35 WSN for environmental monitoring — nodes deployed across a forest to detect fire, measure humidity, and monitor air quality.....63
2.36 WSN in healthcare — wearable sensor nodes monitoring patient vital signs and transmitting data to a medical base station..... 64
2.37 WSN in precision agriculture — soil and climate sensor nodes transmitting data to optimise irrigation and crop management..... 64
2.38 WSN in industrial monitoring — sensor nodes deployed on machinery and infrastructure for predictive maintenance and process control..... 65

2.39 WSN in military applications — covertly deployed sensor nodes providing battlefield surveillance and perimeter monitoring..... 66
2.40 WSN in smart city infrastructure — sensor nodes integrated into urban systems for traffic management, lighting, and public safety..... 67
2.41 Basic components of a Wireless Sensor Network — sensor nodes, base station, communication links, and end-user application.....68
2.42 Internal architecture of a WSN sensor node — sensing, processing, communication, and power subsystems.....70
2.43 WSN base station (sink node) — collecting data from sensor nodes and forwarding it to an external network or application server.....71
2.44 WSN task manager — end-user application receiving, visualising, and acting upon data collected from the sensor network..... 73
2.45 Flat (single-tier) WSN architecture — homogeneous nodes relaying data via multi-hop paths to the base station..... 74
2.46 Hierarchical (clustered) WSN architecture — cluster heads aggregating data from member nodes before forwarding it to the base station.....74
2.47 Hybrid WSN architecture — a combination of flat and hierarchical principles using heterogeneous node roles.....75
2.48 End-to-end data flow in a Wireless Sensor Network, from sensing at the node to delivery at the base station.....76
2.49 WSN protocol stack — physical, data link, network, transport, and application layers, alongside power, mobility, and task management planes.....78
3.50 LabVIEW programming environment showing the Front Panel (left) and Block Diagram (right) of a Virtual Instrument (VI).....87
3.51 LabVIEW DAQ Assistant — a graphical configuration tool for setting up data acquisition tasks from physical hardware..... 89
3.52 LabVIEW as the central monitoring platform in a Wireless Sensor Network architecture, interfacing with distributed sensor nodes.....91
3.53 LabVIEW VISA Serial communication block diagram for reading data from a sensor node via UART.....92
3.54 LabVIEW Block Diagram illustrating the complete data acquisition pipeline for WSN monitoring..... 94

3.55 LabVIEW data logging VI writing timestamped sensor readings to a TDMS file. 95
3.56 LabVIEW Front Panel dashboard for real-time WSN monitoring, displaying waveform charts, numeric indicators, and alarm LEDs.....96
3.57 LabVIEW alarm management block diagram, comparing sensor readings against threshold limits and triggering alarm indicators..... 97
3.58 Three-tier WSN system architecture with LabVIEW at the application layer.....97
3.59 End-to-end data flow from WSN sensor node to LabVIEW monitoring application. 98
3.60 VI hierarchy of the LabVIEW monitoring application developed for this project. 99
3.61 Summary of LabVIEW advantages in WSN monitoring applications.....100
4.62 Logical deployment model of the NI WSN-9791 Gateway with NI WSN-3212 (thermocouple) and NI WSN-3202 (voltage/current) nodes across the PV/T monitoring array..... 105
5.63 LabVIEW 2013 startup screen showing the Create Project option.....127
5.64 Selecting the Blank Project template to begin a new LabVIEW project..... 128
5.65 Newly created WSN_Building.lvproj shown in the Project Explorer.....128
5.66 Adding a new target or device to the LabVIEW project.....129
5.67 Discovering existing targets and devices, including the WSN Gateway..... 129
5.68 Project tree after adding the WSN Gateway and the first sensor node.....130
5.69 Deploying the NI-WSN9791 gateway to establish communication with the hard-ware..... 130
5.70 Deploying the wireless sensor node (Node2, NI WSN-3212)..... 131
5.71 Creating a new VI from the project and the exposed node variables (Battery Voltage, Link Quality, DIO0–DIO3, TC0–TC3, Mesh Router).....131
5.72 Measurement & Automation Explorer (MAX) showing the connected NI WSN-3212 node with Excellent link quality and OK battery state..... 132
5.73 Front panel and block diagram of Building_WSN.vi showing real-time Battery Voltage, Link Quality, and TC0 temperature readings.....133
5.74 Field test setup: the WSN node and thermocouple probe carried outdoors during the range test.....134
5.75 Link Quality of 99 recorded at a distance of approximately 20 cm..... 135
5.76 Link Quality of 50 recorded at a distance greater than 20 m..... 135
5.77 Multi-floor indoor stairwell used for the long-distance link quality test.....136

5.78 Link Quality of 22 recorded at a distance greater than 40 m with intervening walls..... 137
5.79 Link Quality of 17 recorded at a distance greater than 50 m..... 137
5.80 Measurement & Automation Explorer showing both deployed nodes (NI WSN-3212 and NI WSN-3202) with Excellent link quality..... 138
5.81 Block diagram reading Battery Voltage, Link Quality, TC0, TC1, Battery Voltage 2, Link Quality 2, and AI0 from both nodes..... 139
5.82 Front panel layout for the two-node system: Node NI WSN-3212 (Thermocouple Sensor) and Node NI WSN-3202 (Soil Moisture Sensor)..... 140
5.83 Live operation of the two-node system showing TC0 = 56.08 °C, TC1 = 25.45 °C, and a soil moisture reading of AI0 = 0.949, together with the corresponding waveform chart..... 140
5.84 Laboratory setup of the complete two-node WSN system, including the gateway, both sensor nodes, the soil moisture sensor board, the thermocouple probe, and the host laptop running LabVIEW..... 141
5.85 Block diagram of Host Controller VI.vi, showing the temperature comparison logic that triggers the high-temperature alarm and invokes the Telegram alert Python script..... 142
5.86 Host Controller VI front panel during normal operation (TC0 = 31.33 °C, High Temp Alarm indicator OFF)..... 143
5.87 Host Controller VI front panel with the High Temp Alarm indicator active, together with the live temperature trend on the waveform chart..... 144
5.88 Python `telegram_alert.py` script in Visual Studio Code — implementing Telegram Bot API notification with status-file-based duplicate suppression and an audible beep alert via `winsound`..... 145
5.89 Telegram notification received on the WSN monitoring group, reporting a high-temperature reading of 54.76 °C against the 35 °C threshold..... 146

List of Tables

1.3 Comparison of Wired and Wireless Communication.....52
1.4 Infrastructure-based vs Ad Hoc Networks.....55
4.5 Reference hardware parameters used in the problem formulation.....103
4.6 Illustrative transmission energy per packet at varying node-to-gateway distance ($l = 512$ bits).....111
4.7 Mapping between abstract decision variables and their LabVIEW/NI WSN implementation.....119
4.8 Assumed parameter values for the numerical case study.....120
4.9 Summary of design trade-off resolutions adopted for the PV/T WSN.....124
4.10 Summary of notation used in the WSN problem formulation.....125
5.11 Summary of distance versus link quality test results.....138

List of Parameters

Symbol	Description	Unit
Physical Measurement Quantities		
T	Temperature	°C
V	Voltage (general)	V
I	Current (general)	A
P	Power	W
V_{SA}	Solar array voltage	V
V_L	Load voltage	V
I_{SA}	Solar array current	A
I_L	Load current	A
I_B	Battery current	A
$Insol$	Insolation (solar irradiance)	W/m²
SOC	State of charge of the battery	%
Network Topology and Node Notation (Chapter 4)		
$G = (V, E)$	Network topology graph; nodes V and feasible communication links E	—
n_i, GW	Sensor node i; the NI WSN-9791 gateway node	—
p_i	Spatial coordinates of node n_i	m
$E_i(t)$	Residual battery energy of node n_i at time t	J
$s_i(t)$	Binary active/sleep state of node n_i	—
τ_i	Node hardware type (3212 thermocouple or 3202 voltage/current)	—
C_T, C_E	Thermal and electrical sensor clusters	—
A, L	Adjacency matrix and graph Laplacian of G	—

Symbol	Description	Unit
k, k_{max}	Number of deployed nodes; maximum node count supported by gateway	—
Network Lifetime Metrics (Chapter 4)		
T_{life}	General network lifetime metric	s
T_{HND}	Half Node Death lifetime	s
T_{LND}	Last Node Death lifetime	s
T_{cov}	Coverage-preserving network lifetime	s
Energy Model (Chapter 4)		
E_{i}^{total}	Total per-node energy consumption	J
E_{i}^{sense}	Per-node sensing energy	J
E_{i}^{proc}	Per-node processing energy	J
E_{i}^{tx}	Per-node transmit energy	J
E_{i}^{rx}	Per-node receive energy	J
e_{ADC}	Energy per ADC sampling cycle	J
e_{CJC}	Cold-junction-compensation energy	J
E_{elec}	Radio electronics energy coefficient	J/bit
ε_{fs}, ε_{mp}	Free-space and multi-path radio energy coefficients	J/bit/m², J/bit/m⁴
d_0	Crossover distance of the radio energy model	m
E_{max}	Maximum battery capacity	J
E_{th}	Minimum operational energy threshold	J
δ_{max}	Maximum permissible duty cycle	—

Symbol	Description	Unit
$P_{col}(k)$	Probability of channel-access collision among k nodes	—
PDR	Packet Delivery Ratio	%
$N_{received}$, N_{sent}	Number of packets received; number of packets sent	—
$\bar{L}$	Average end-to-end latency	s
R	Composite reliability objective	—
w_1, w_2	Weighting coefficients of the composite objective	—
$P_{success}(n_i)$	End-to-end path success probability for node n_i	—
L_{alert}	Alert packet latency	s
L_{alert}^{max}	Required upper bound on alert latency	s

Symbol	Description	Unit
Sampling and Timing (Chapter 4)		
Δt_T	Thermal sampling interval (WSN-3212)	s
Δt_E	Electrical sampling interval (WSN-3202)	s
Δt_{min}, Δt_{max}	Lower and upper bounds on sampling interval	s
τ_{th}	Thermal time constant of the PV/T absorber plate	s
Decision Variables (Chapter 4)		
r_i	Routing path decision variable for node n_i	—
f_i	Sampling frequency decision variable for node n_i	Hz
Communication and Reliability (Chapter 4)		
$R_{\max}$	Maximum transmission range	m
$RSSI_{min}$	Minimum receiver sensitivity	dBm
$H(n_i)$	Hop count of node n_i to the gateway	—
H_{max}	Maximum permitted hop count	—
C_{ch}	Effective wireless channel capacity	bps
$\sigma(t)$	Transmission schedule of node n_i	—
σ_{alert}	Reserved alert transmission slot	—

Chapter 1

Introduction of Networks

Introduction

Computer networks have become a fundamental pillar of modern life, enabling seamless communication, resource sharing, and data exchange across the globe. From small home networks to massive enterprise infrastructures, networks connect billions of devices and users, forming the backbone of the digital world. This chapter introduces the core concepts of computer networks, including their definitions, components, types, and topologies, providing the foundational knowledge required to understand how data travels across interconnected systems.

1.1 Network Definition

A computer network is a group of interconnected computing devices that communicate with one another to share resources, exchange data, and provide services. These devices, known as **nodes**, include computers, servers, switches, routers, and other hardware components that are linked using wired or wireless media.

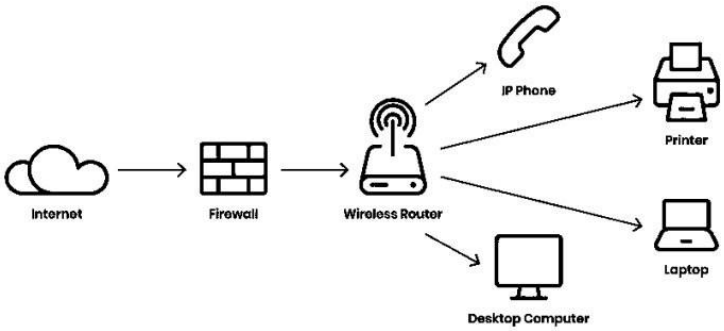

Figure 1.1: *A typical computer network showing Internet, Firewall, Wireless Router, and connected end devices.*

Regardless of the scale, computer networks can be leveraged for many purposes, such as:

- Digital communication via email, instant messaging, etc.
- Connecting devices such as printers, scanners and photocopiers.
- Sharing files via wireless communication.
- Sharing and operating software on remote workstations.
- Hosting and accessing virtual machines and containers.

1.2 Network Components

1.2.1 End Devices

End devices are the source or destination of data in a network. Common end devices include:

- Personal Computer (PC)
- Laptop
- Smart Phone
- Network Printer

1.2.2 Intermediate Devices

Intermediate devices connect individual hosts to the network and interconnect multiple networks. They manage data flow by determining the path that messages take through the network. Common intermediate devices include:

- **Switch** — connects multiple devices within the same network (LAN) and forwards data only to the intended recipient using MAC addresses.
- **Router** — connects different networks together and directs data packets between them using IP addresses.
- **Wireless Access Point (WAP)** — allows wireless devices to connect to a wired network using Wi-Fi.
- **Firewall** — monitors and controls incoming and outgoing network traffic based on predefined security rules.
- **Modem** — modulates and demodulates signals to enable communication between a local network and an Internet Service Provider (ISP).

1.2.3 Network Media

Network media is the physical or wireless medium through which data travels between devices. The three main types are:

- **Copper cable (Ethernet)** — transmits data as electrical signals; the most common type is the Unshielded Twisted Pair (UTP) cable, used in most LANs due to its low cost and ease of installation.
- **Fiber-optic cable** — transmits data as pulses of light; supports very high speeds over long distances with no electromagnetic interference.
- **Wireless media** — transmits data using radio waves (Wi-Fi, Bluetooth, cellular); provides mobility and flexibility but is more susceptible to interference and security threats.

1.2.4 Network Services and Protocols

Network services and protocols define the rules and standards that govern how data is formatted, transmitted, and received across a network. Key protocols include:

- **TCP/IP** (Transmission Control Protocol / Internet Protocol) — the foundational protocol suite of the internet.
- **HTTP/HTTPS** — used for web communication; HTTPS adds encryption via TLS/SSL.
- **DNS** (Domain Name System) — translates human-readable domain names (e.g., www.example.com) into IP addresses.
- **DHCP** (Dynamic Host Configuration Protocol) — automatically assigns IP addresses to devices joining the network.

1.3 Computer Network Types

1.3.1 Local Area Networks (LANs)

The group of devices that connect together in a small area (small geographic area), such as a building floor, a building, or adjacent buildings; this is called a **Local Area Network (LAN)**. The advantage of a LAN is that all network elements are close together, which results in high data transfer speeds, low latency, and reduced communication costs. LANs are commonly used in homes, offices, schools, and campuses.

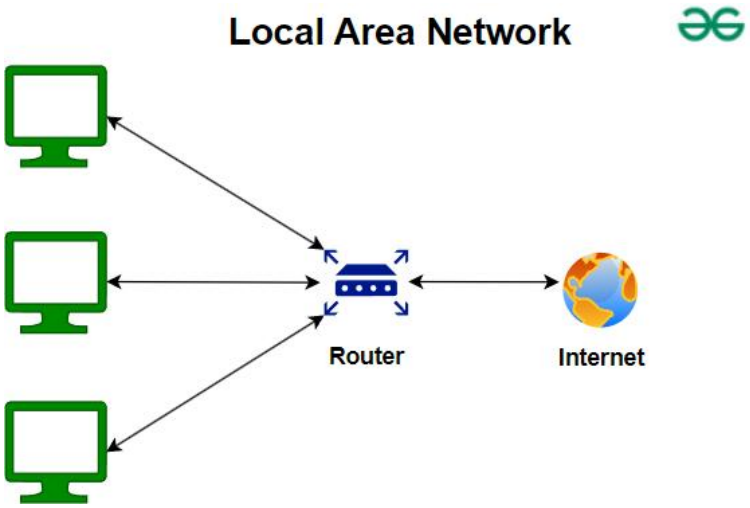

Figure 1.2: *Local Area Network (LAN) — multiple devices connected through a router to the Internet.*

1.3.2 Metropolitan Area Networks (MANs)

A **Metropolitan Area Network (MAN)** is a network that spans a city or a large campus, covering a geographic area larger than a LAN but smaller than a WAN. MANs are typically owned and operated by a single organization or an Internet Service Provider (ISP) and are used to interconnect multiple LANs within the same metropolitan region.

Key characteristics of a MAN include:

- Covers a geographic range of approximately 5 to 50 kilometres.
- Higher data transfer speeds compared to WANs.
- Commonly implemented using fiber-optic cables or wireless links.
- Used by city governments, universities, and large enterprises to connect multiple buildings or campuses.
- Provides a shared network infrastructure that reduces the cost of individual long-distance connections.

A typical example of a MAN is a cable television network within a city, or a network connecting multiple university campuses across an urban area.

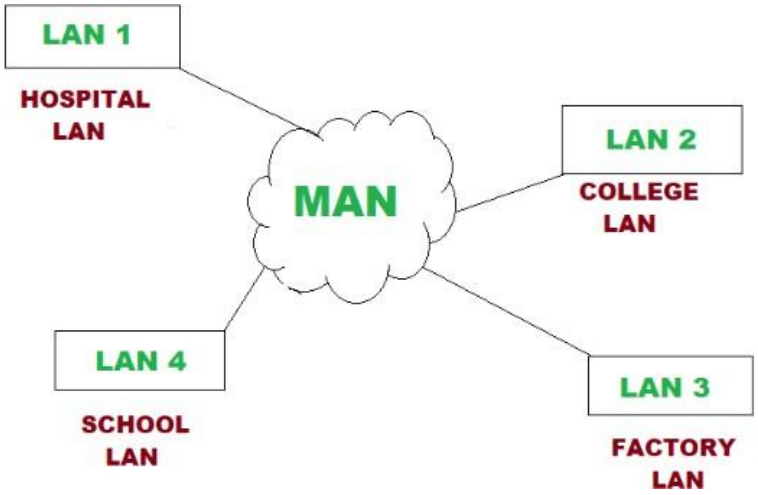

Figure 1.3: *Metropolitan Area Network (MAN) — multiple LANs (Hospital, College, Factory, School) interconnected through a central MAN.*

1.3.3 Wide Area Networks (WANs)

A **Wide Area Network (WAN)** is a network that covers a broad geographic area, such as a country, a continent, or even the entire globe. A WAN connects multiple LANs and MANs together over long distances, typically using leased telecommunication lines, fiber-optic cables, or satellite links. The most well-known example of a WAN is the **Internet**.

Key characteristics of a WAN include:

- Covers an unlimited geographic range — from hundreds to thousands of kilometres.
- Connects multiple LANs and MANs across different cities and countries.
- Typically owned and managed by telecommunication providers or ISPs.
- Lower data transfer speeds compared to LANs due to the long distances involved.
- Uses technologies such as MPLS, leased lines, DSL, fiber optics, and satellite communication.
- Higher cost of implementation and maintenance compared to LANs and MANs.

WANs are essential for large organizations, multinational companies, and government institutions that need to share data and resources across geographically distributed locations. For example, a bank with branches in multiple countries relies on a WAN to synchronize transactions and data in real time.

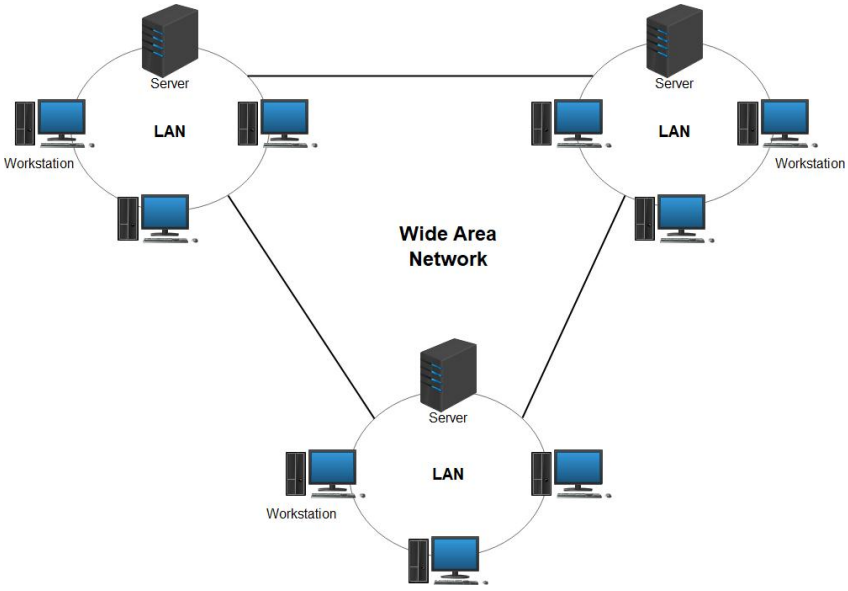

Figure 1.4: *Wide Area Network (WAN) — multiple LANs interconnected across geographically distant locations through dedicated server links.*

1.4 Network Topology

Network topology refers to the physical or logical arrangement of devices and connections in a network. It describes how different nodes are interconnected and how data flows between them. Choosing the right topology is a critical design decision, as it directly affects the network's performance, scalability, fault tolerance, and cost. The main types of network topologies are: Point-to-Point, Bus, Ring, Star, Mesh, and Tree.

1.4.1 Point-to-Point Topology

A **Point-to-Point (P2P)** topology is the simplest form of network topology, in which a dedicated communication link connects exactly two devices directly to each other without any intermediate device. Each device has full use of the entire link's bandwidth, resulting in fast and reliable communication between the two endpoints.

Key characteristics of Point-to-Point topology include:

- Dedicated link between exactly two nodes.
- Maximum bandwidth availability since no sharing occurs.
- Very low latency due to the direct connection.
- Simple to configure and implement.
- Limited scalability — only two devices can communicate per link.

Point-to-Point connections are widely used in:

- Telephone calls between two parties.
- A direct connection between a computer and a printer.
- WAN links connecting two remote routers (e.g., leased lines).
- Virtual Private Network (VPN) tunnels between two endpoints.

Figure 1.5: *Point-to-Point Topology — a dedicated link connecting two devices directly.*

1.4.2 Bus Topology

In a **Bus topology**, all devices are connected to a single shared communication medium, known as the **bus** or **backbone cable**. Data transmitted by any device travels along the entire length of the bus and is received by all connected devices; however, only the intended recipient processes the data. The bus is terminated at both ends to prevent signal reflection.

Key characteristics of Bus topology include:

- All devices share a single communication medium (typically coaxial cable or twisted pair).
- Simple and inexpensive to install for small networks.
- A single collision domain — only one device can transmit at a time, which increases the probability of collisions as the number of devices grows.
- A failure in the backbone cable brings down the entire network.
- Difficult to troubleshoot and not scalable for large networks.

Due to its limitations in performance and fault tolerance, Bus topology is largely considered obsolete in modern networks and has been replaced by Star and Switch-based topologies.

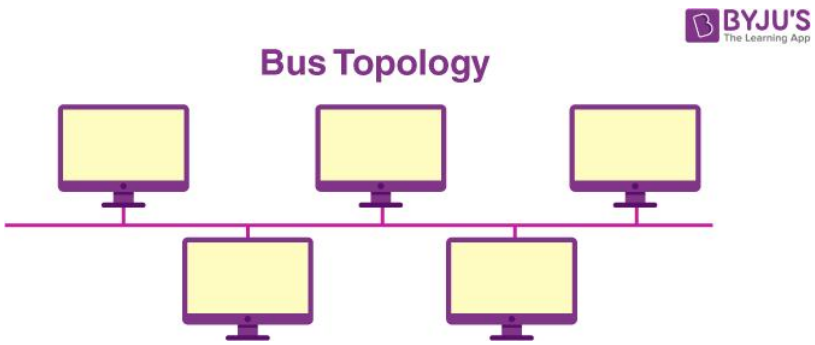

Figure 1.6: *Bus Topology — all devices connected to a single shared backbone cable.*

1.4.3 Ring Topology

In a **Ring topology**, each device is connected to exactly two other devices, forming a closed circular loop. Data travels in one direction (unidirectional) or both directions (bidirectional in a dual-ring setup) around the ring until it reaches its intended destination.

Key characteristics of Ring topology include:

- Each device is connected to two neighbours, forming a continuous loop.
- Data travels in a single direction, reducing the chance of collisions.
- Each device acts as a repeater, regenerating the signal as it passes through.
- A failure in any single device or cable can disable the entire network (unless a dual-ring is used).
- Requires more cable than Bus topology but less than Mesh topology.
- Commonly used in Token Ring and FDDI (Fiber Distributed Data Interface) networks.

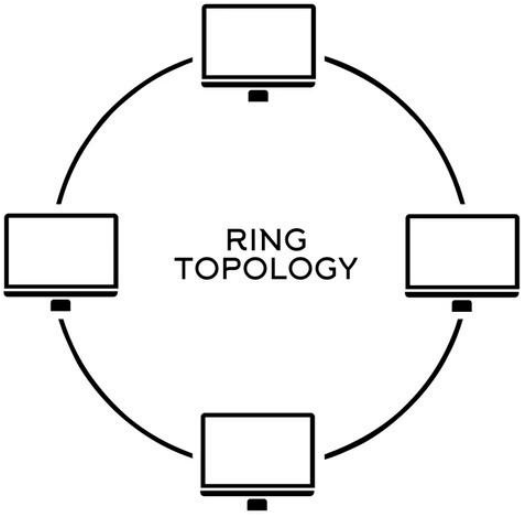

Figure 1.7: *Ring Topology — devices connected in a closed circular loop.*

1.4.4 Mesh Topology

In a **Mesh topology**, every device is connected to every other device in the network through dedicated point-to-point links. This creates multiple redundant paths for data to travel, ensuring high fault tolerance and reliability.

Key characteristics of Mesh topology include:

- Every node has a dedicated link to every other node.
- Provides maximum redundancy — if one link fails, data is rerouted through an alternative path.
- Highly fault tolerant and reliable, making it suitable for mission-critical networks.
- Very expensive to implement due to the large number of cables and interfaces required.
- The number of links required for a full mesh of n nodes is $n(n-1)2$.
- A **partial mesh** is a cost-effective compromise, where only some devices are fully interconnected.

Mesh topology is commonly used in the WAN core, military communications, and backbone infrastructure where reliability is paramount.

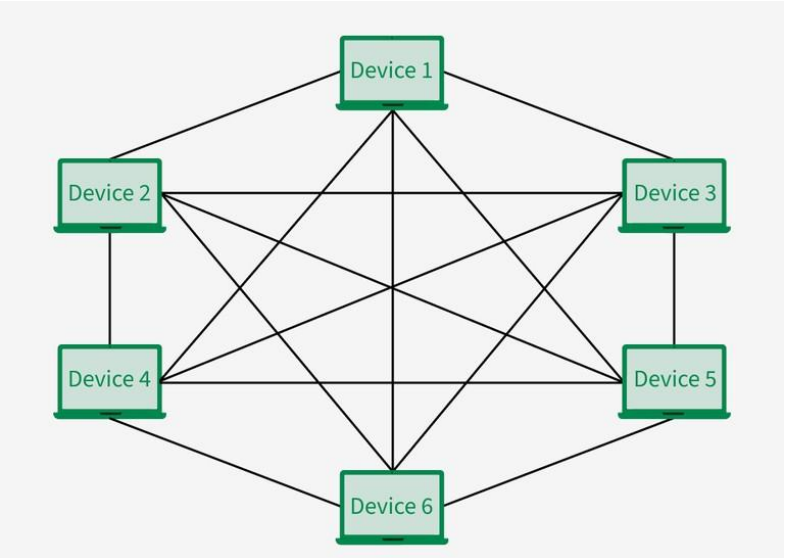

Figure 1.8: *Mesh Topology — every device connected to every other device through dedicated links.*

1.4.5 Star Topology

In a **Star topology**, all devices are connected to a central device, typically a **switch** or **hub**. All communication between devices passes through this central node, making it the most widely deployed topology in modern LANs.

Key characteristics of Star topology include:

- All devices connect to a central switch or hub.
- Easy to install, manage, and troubleshoot — each device has its own dedicated cable to the central node.
- A failure in one cable or device does not affect the rest of the network.
- If the central switch fails, the entire network goes down.
- Less cabling required compared to Mesh topology.
- Supports easy addition or removal of devices without disrupting the network.
- Low collision rate when using a switch, as each port forms its own collision domain.

Star topology is the standard choice for home networks, offices, schools, and enterprise LANs due to its simplicity, reliability, and ease of management. It is the topology adopted in the network design of this project.

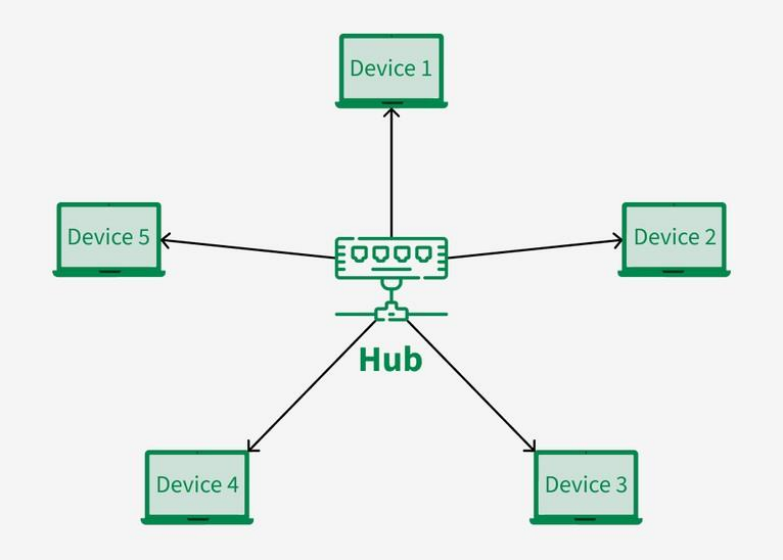

Figure 1.9: *Star Topology — all devices connected to a central switch.*

1.5 Network Model

A **network model** is a conceptual framework that defines how data is transmitted, formatted, and processed across a network. Network models provide a standardised structure that allows different hardware and software systems from various vendors to communicate with one another. The two most important and widely used network models are the **OSI model** and the **TCP/IP model**.

1.5.1 Open Systems Interconnection (OSI) Model

The **OSI (Open Systems Interconnection)** model is a conceptual framework developed by the International Organization for Standardization (ISO). It organises network communication into **seven distinct layers**, each with specific responsibilities. The OSI model is primarily used today as a **reference model** to help engineers understand, design, and troubleshoot networks in a structured and vendor-neutral way.

Each layer communicates with the layer directly above and below it, and data flows down through the layers on the sender side and back up on the receiver side.

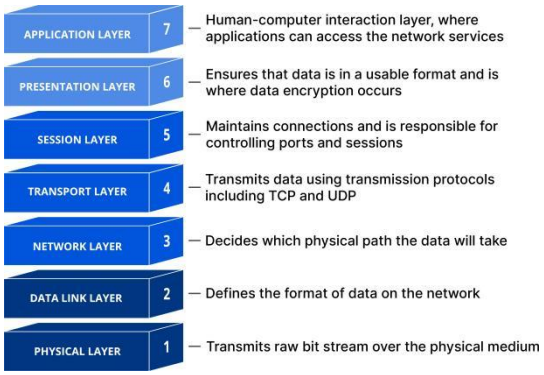
APPLICATION LAYER
7
Human-computer interaction layer, where applications can access the network services
PRESENTATION LAYER
6
Ensures that data is in a usable format and is where data encryption occurs
SESSION LAYER
5
Maintains connections and is responsible for controlling ports and sessions
TRANSPORT LAYER
4
Transmits data using transmission protocols including TCP and UDP
NETWORK LAYER
3
Decides which physical path the data will take
DATA LINK LAYER
2
Defines the format of data on the network
PHYSICAL LAYER
1
Transmits raw bit stream over the physical medium

Figure 1.10: *The Seven Layers of the OSI Model.*

Layer 7 — Application Layer

The **Application layer** is the topmost layer of the OSI model and serves as the interface between the network and the end user. It is responsible for preparing the initial request or reply for a service and providing network services directly to user applications such as web browsers, email clients, and file transfer tools.

Key protocols at this layer include:

- **HTTP** (HyperText Transfer Protocol) — allows access to data over the World Wide Web; transmits data in plain text, which is a security disadvantage. It is recommended to use **HTTPS** instead, which encrypts the transmitted data using TLS/SSL.
- **FTP** (File Transfer Protocol) — used to transmit and receive files between a client and a server over a network.
- **POP3** (Post Office Protocol v3) — used to receive emails from a mail server to a local client.
- **SMTP** (Simple Mail Transfer Protocol) — used to send emails from a client to a mail server or between mail servers.
- **DNS** (Domain Name System) — resolves human-readable domain names into IP addresses, making communication simpler for users.
- **Telnet** — provides remote login capability, allowing a user to access and manage a device over the network.

Figure 1.11: *OSI Layer 7 — Application Layer protocols and services.*

Layer 6 — Presentation Layer

The **Presentation layer**, often referred to as the **syntax layer** or **translation layer**, acts as a translator between the application layer and the rest of the network. It ensures that data sent by one application can be read and understood by the receiving application, regardless of differences in data representation between systems.

The main functions of the Presentation layer include:

- **Data Translation** — converts data from the application format into a common network format and vice versa.
- **Encryption and Decryption** — encrypts data before transmission to ensure confidentiality; SSL/TLS operates at this layer.
- **Compression and Decompression** — reduces the size of data to improve transmission efficiency.
- **File Formatting** — adds the appropriate file extension (e.g., .jpg, .mp4, .pdf) so the receiving application knows how to interpret the data.

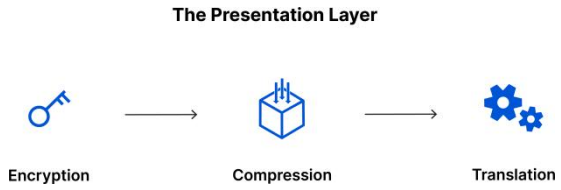

Figure 1.12: *OSI Layer 6 — Presentation Layer: encryption, compression, and translation.*

Layer 5 — Session Layer

The **Session layer** is responsible for establishing, managing, and terminating communication sessions between two devices. This layer ensures that data exchanges are reliable, organised, and synchronised.

The key functions of the Session layer include:

- **Session Establishment** — sets up a session by coordinating and negotiating parameters such as protocols, port numbers, and authentication.
- **Session Maintenance** — keeps the session alive during data transfer, handling interruptions and re-establishing the connection if necessary.
- **Session Termination** — gracefully closes the session once the data exchange is complete, freeing up network resources.
- **Synchronisation** — inserts checkpoints into the data stream so that transmission can resume from the last checkpoint in the event of a failure.
- **Dialog Control** — manages whether communication is half-duplex or full-duplex.

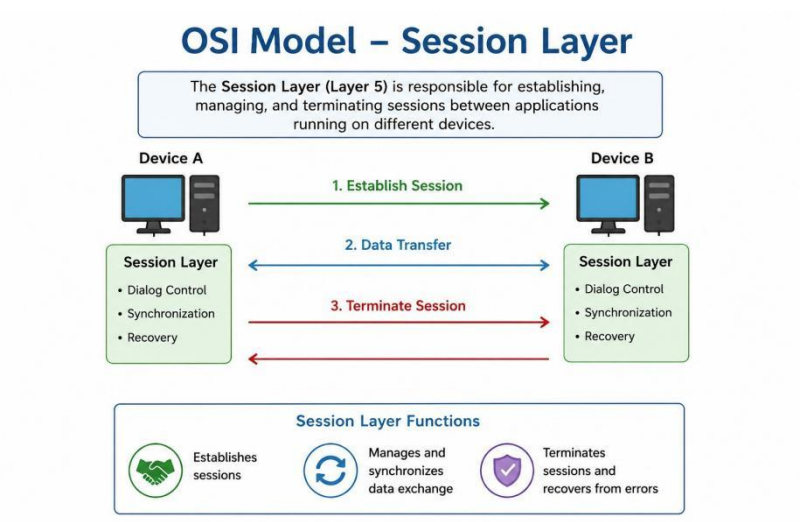

Figure 1.13: *OSI Layer 5 — Session Layer: establishing and managing communication sessions.*

Layer 4 — Transport Layer

The **Transport layer** (Layer 4) is responsible for end-to-end communication between applications on different hosts. The data unit at this layer is called a **segment**.

The key functions of the Transport layer include:

- **Segmentation and Reassembly** — breaks large data into smaller segments for transmission and reassembles them at the destination.
- **Flow Control** — regulates the rate of data transmission to prevent the sender from overwhelming the receiver.
- **Error Control** — detects and requests retransmission of lost or corrupted segments.
- **Port Addressing** — uses port numbers to identify specific applications (e.g., port 80 for HTTP, port 443 for HTTPS).

The Transport layer has two main protocols:

- **TCP** (Transmission Control Protocol) — connection-oriented, reliable, and ordered. Uses a three-way handshake (SYN, SYN-ACK, ACK) to establish connections. Suitable for web browsing, email, and file transfer.
- **UDP** (User Datagram Protocol) — connectionless, faster, and less reliable. Ideal for video streaming, online gaming, and VoIP where speed is more important than guaranteed delivery.

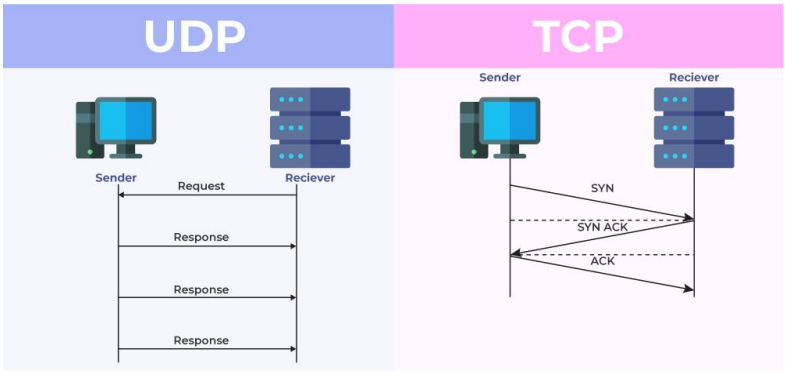

Figure 1.14: *OSI Layer 4 — Transport Layer: TCP vs UDP and segmentation process.*

Layer 3 — Network Layer

The **Network layer** (Layer 3) is responsible for logical addressing and routing. The data unit at this layer is called a **packet**.

The key functions of the Network layer include:

- **Logical Addressing** — assigns and manages IP addresses (IPv4 or IPv6) to uniquely identify devices across different networks.
- **Routing** — determines the optimal path for packets using routing protocols such as OSPF, EIGRP, and BGP.
- **Packet Forwarding** — moves packets from one network to another through intermediate routers.
- **Fragmentation and Reassembly** — breaks packets into smaller fragments if they exceed the MTU of a network, and reassembles them at the destination.

The primary device operating at this layer is the **Router**.

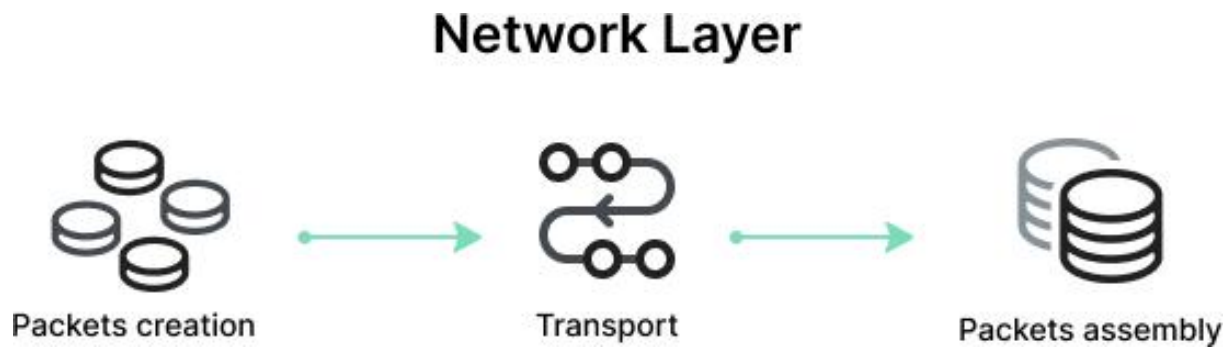

Figure 1.15: *OSI Layer 3 — Network Layer: IP addressing and packet routing.*

Layer 2 — Data Link Layer

The **Data Link layer** (Layer 2) is responsible for reliable data transfer between two directly connected nodes. The data unit at this layer is called a **frame**.

The Data Link layer is divided into two sublayers:

- **LLC** (Logical Link Control) — provides an interface between the Network layer and the MAC sublayer.
- **MAC** (Media Access Control) — controls how devices access the shared medium and permission to transmit data.

The key functions of the Data Link layer include:

- **Framing** — encapsulates packets into frames by adding MAC address headers and error-detection trailers.
- **Physical Addressing** — uses MAC addresses to identify sender and receiver on the same local network.
- **Error Detection** — uses CRC (Cyclic Redundancy Check) to detect errors in transmitted frames.
- **Flow Control** — prevents a fast sender from overwhelming a slow receiver on the same link.
- **Access Control** — determines which device has the right to use the shared medium at a given time.

The primary device operating at this layer is the **Switch**.

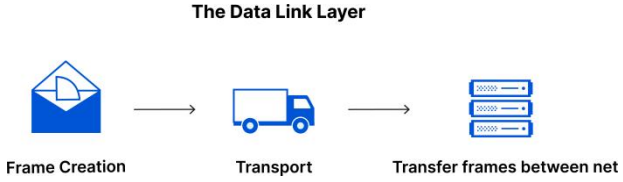

Figure 1.16: *OSI Layer 2 — Data Link Layer: MAC addressing and frame structure.*

Layer 1 — Physical Layer

The **Physical layer** is the lowest layer of the OSI model, responsible for the actual transmission of raw **bits** (0s and 1s) over a physical medium such as copper cables, fiber-optic cables, or wireless radio signals.

The key functions of the Physical layer include:

- **Bit Transmission** — converts digital bits into physical signals (electrical voltages, light pulses, or radio waves) for transmission, and converts received signals back into bits.
- **Physical Medium Specifications** — defines the type of cable, connector, and signal encoding (e.g., RJ-45, fiber-optic, coaxial).
- **Data Rate Control** — defines the transmission speed in bits per second (bps).
- **Synchronisation** — ensures sender and receiver are synchronised at the bit level.
- **Topology and Line Configuration** — defines how devices are physically connected (point-to-point or multipoint).

Devices operating at this layer include **hubs**, **repeaters**, **cables**, and **network interface cards (NICs)**.

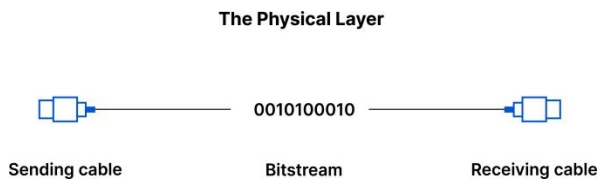

Figure 1.17: *OSI Layer 1 — Physical Layer: bit transmission over physical media.*

1.5.2 TCP/IP Model

The **TCP/IP model**, also known as the **Internet Protocol Suite**, is the practical framework that governs all communication on the modern internet. It was developed in the 1970s by Vinton Cerf and Bob Kahn under DARPA and was originally the standard protocol for ARPANET. Unlike the OSI model, which has seven layers, the TCP/IP model consolidates communication into **four layers**:

- **Application Layer** — combines OSI layers 5, 6, and 7. Handles protocols such as HTTP, HTTPS, FTP, DNS, SMTP, and DHCP.
- **Transport Layer** — corresponds to OSI Layer 4. Provides end-to-end communication using TCP or UDP.
- **Internet Layer** — corresponds to OSI Layer 3. Handles logical IP addressing and routing using IPv4 or IPv6.
- **Network Interface Layer** — combines OSI layers 1 and 2. Manages physical transmission and MAC-level framing using Ethernet and Wi-Fi.

TCP/IP is the foundation of the internet and the protocol suite implemented in this project. TCP ensures reliable delivery of application data, while UDP is used where low latency is more critical than guaranteed delivery.

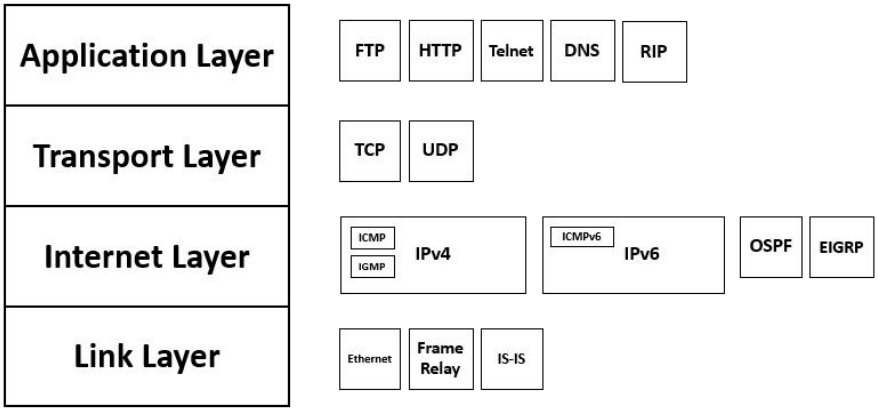

Figure 1.18: *TCP/IP Model layers compared to the OSI Model.*

1.6 Network Devices

Network devices are the physical components that form the infrastructure of a computer network. They are responsible for establishing and maintaining communication pathways, ensuring data is transmitted and received securely and efficiently. Network devices can be classified into basic devices (Hub, Switch, Router, Modem, NIC) and advanced devices (Firewall, IDS, IPS, Wireless Access Point).

1.6.1 Hub

A **hub** is one of the simplest networking devices, operating at **Layer 1 (Physical layer)** of the OSI model. When a hub receives a data packet on one of its ports, it broadcasts that packet to **all** other connected ports regardless of the intended destination, which wastes bandwidth and creates unnecessary traffic.

Key characteristics of a Hub:

- Operates at OSI Layer 1 (Physical layer).
- Broadcasts data to all connected ports with no intelligence to direct traffic to a specific device.
- Creates a single collision domain — only one device can transmit at a time.
- Half-duplex operation — cannot send and receive simultaneously.
- No MAC address table.
- Using a hub in a modern network is **not recommended**.

Hubs come in three types:

- **Active Hub** — amplifies and regenerates the incoming signal before retransmitting. Also known as a *multiport repeater*.
- **Passive Hub** — relays signals without amplification. Does not require a power source.
- **Intelligent Hub** — adds network monitoring, traffic analysis, and device isolation capabilities.

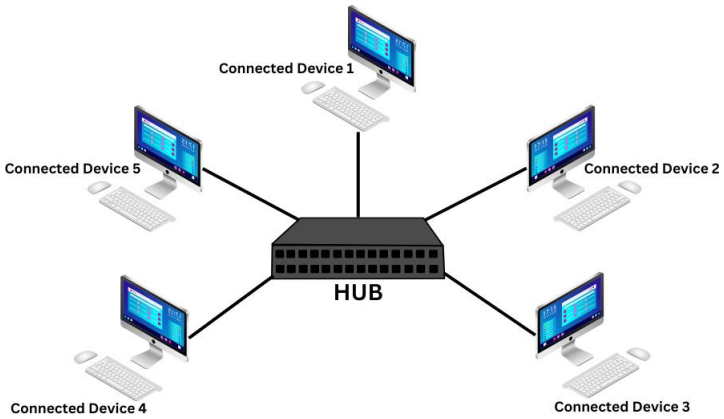

Figure 1.19: *Network Hub — broadcasts data to all connected devices.*

1.6.2 Repeater

A **repeater** operates at **Layer 1 (Physical layer)** of the OSI model. Its primary function is to receive a weakened or degraded signal, regenerate and amplify it, and retransmit it over a greater distance.

Key characteristics of a Repeater:

- Regenerates and amplifies signals to extend network range.
- Does not filter or interpret data — it simply regenerates raw bits.
- Transparent to higher-layer protocols.
- Cannot connect networks that use different protocols or speeds.

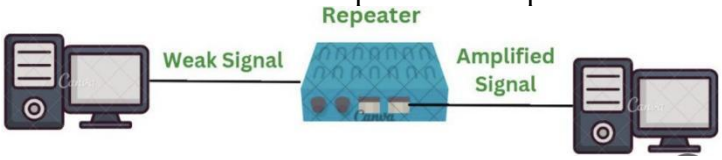

Figure 1.20: *Repeater — regenerates and amplifies signals to extend network coverage.*

1.6.3 Bridge

A **bridge** operates at **Layer 2 (Data Link layer)** of the OSI model. It connects two or more network segments and filters traffic based on **MAC addresses**, forwarding frames only to the relevant segment.

Key characteristics of a Bridge:

- Learns MAC addresses and builds a MAC address table.
- Reduces collision domains by segmenting the network.
- Forwards frames only to the segment where the destination device is located.
- Multiple LANs can be connected through a bridge to form a larger extended LAN.

There are two main types of bridges:

- **Transparent Bridge** — invisible to end devices; automatically learns MAC-to-port mappings. Used in Ethernet environments.
- **Source Routing Bridge** — the sending device specifies the complete path in the frame header. Used in Token Ring networks.

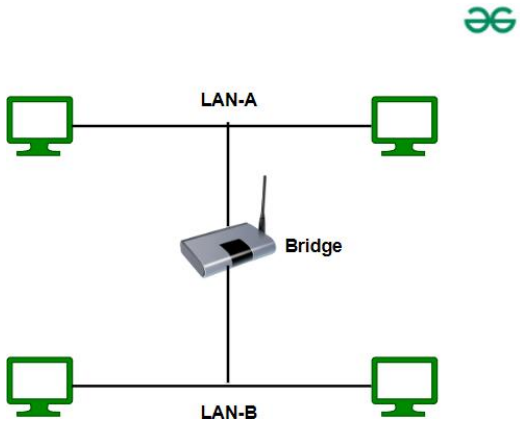

Figure 1.21: *Bridge — connects two network segments and filters traffic using MAC addresses.*

1.6.4 Switch

A **switch** operates primarily at **Layer 2 (Data Link layer)** of the OSI model, although advanced switches also operate at Layer 3. It intelligently forwards data frames only to the intended recipient based on MAC addresses, unlike a hub which broadcasts to all ports.

Key characteristics of a Switch:

- Maintains a **MAC address table** to map MAC addresses to specific ports.
- Each port forms its own **collision domain**, eliminating collisions.
- Supports **full-duplex** communication on each port.
- Supports **VLANs** to logically segment a network.
- Significantly more efficient and secure than a hub.

Common types of switches include:

- **Unmanaged Switch** — plug-and-play; no configuration required. Suitable for home or small-office networks.
- **Managed Switch** — fully configurable; supports VLANs, QoS, STP, port mirroring, and SNMP monitoring.
- **Layer 3 Switch (Multilayer Switch)** — combines L2 switching with L3 routing; can route between VLANs.
- **PoE Switch** — delivers data and electrical power over Ethernet cables to IP cameras, access points, and VoIP phones.
- **Gigabit Switch** — supports speeds up to 1 Gbps per port.

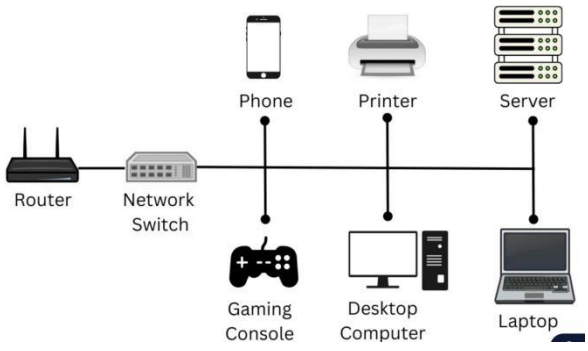

Figure 1.22: *Network Switch — intelligently forwards data to the intended recipient using MAC addresses.*

1.6.5 Router

A **router** operates at **Layer 3 (Network layer)** of the OSI model. It forwards data packets between different networks by determining the best path for each packet based on **IP addresses**.

Key characteristics of a Router:

- Maintains a **routing table** to determine the optimal path for packet forwarding.
- Connects multiple different networks (LAN to WAN or LAN to internet).
- Performs **NAT** to allow multiple devices to share a single public IP address.
- Can act as a **DHCP server**, assigning IP addresses dynamically.
- Provides basic **firewall** protection by filtering packets.
- Supports routing protocols such as OSPF, EIGRP, and BGP.

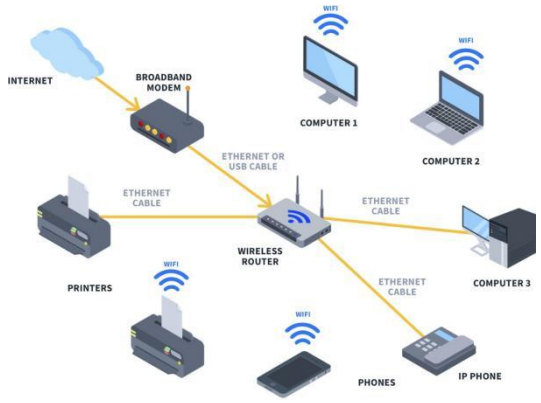

Figure 1.23: *Router — forwards packets between networks and connects LANs to the internet.*

1.6.6 Modem

A **modem** (MOdulator-DEModulator) connects a local network to the internet by converting digital signals into a format suitable for transmission over the ISP's infrastructure and vice versa.

Key characteristics of a Modem:

- Modulates outgoing digital signals into analog format for the ISP medium.
- Demodulates incoming analog signals back into digital format.
- Connects via DSL, cable, fiber, or cellular.
- Often combined with a router in a single device in home networks.

- Assigned a public IP address by the ISP.

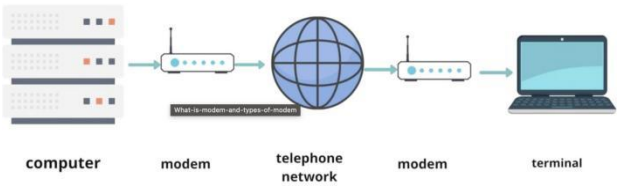

Figure 1.24: *Modem — bridges a local network to the internet via the ISP infrastructure.*

1.6.7 Network Interface Card (NIC)

A **Network Interface Card (NIC)** is a hardware component that enables a computing device to connect to a network. It operates at both **Layer 1 (Physical)** and **Layer 2 (Data Link)** of the OSI model.

Key characteristics of a NIC:

- Each NIC has a globally unique **MAC address** (48-bit) burned into its firmware at the factory.
- Available as wired (Ethernet) or wireless (Wi-Fi) variants.
- Modern computers often have the NIC integrated on the motherboard.
- Supports various speeds: 10 Mbps, 100 Mbps, 1 Gbps, 10 Gbps.

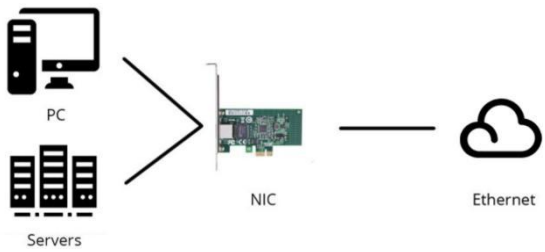

Figure 1.25: *Network Interface Card (NIC) — connects a device to the network.*

1.6.8 Firewall

A **firewall** monitors and controls incoming and outgoing network traffic based on predefined security rules. It acts as a barrier between a trusted internal network and untrusted external networks such as the internet.

Key characteristics of a Firewall:

- Filters traffic based on IP addresses, port numbers, and protocols.
- Can be hardware-based, software-based, or both.
- **Stateful Firewall** — tracks active connections and makes filtering decisions based on traffic context.
- **Next-Generation Firewall (NGFW)** — combines traditional features with deep packet inspection (DPI), intrusion prevention, and application awareness.
- Prevents DDoS, port scanning, IP spoofing, and unauthorised access.

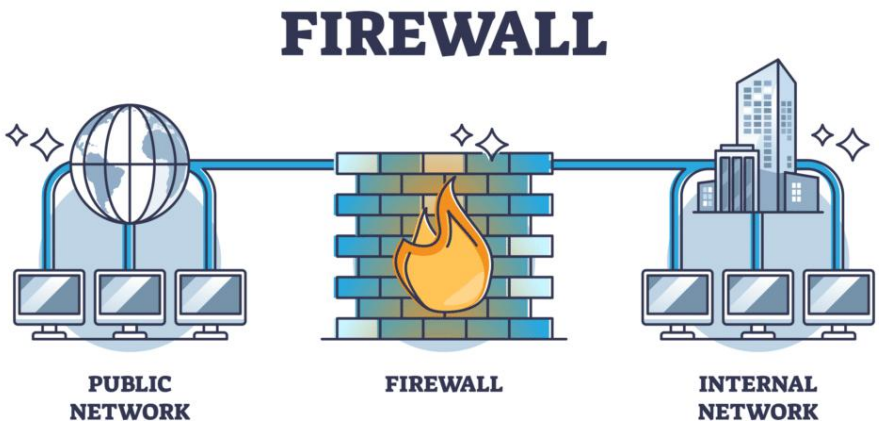

Figure 1.26: *Firewall — monitors and filters network traffic to protect the internal network.*

1.6.9 Wireless Access Point (WAP)

A **Wireless Access Point (WAP)** allows wireless devices to connect to a wired network using Wi-Fi (IEEE 802.11). It operates at **Layer 2 (Data Link layer)** of the OSI model.

Key characteristics of a WAP:

- Transmits and receives data using radio frequency signals (2.4 GHz, 5 GHz, or 6 GHz).
- Identified by an **SSID** — the network name visible to wireless devices.
- Supports WPA2 and WPA3 security protocols to encrypt wireless communication.
- Can be powered via PoE from a PoE switch.
- Differs from a wireless router — a WAP only provides wireless access, while a router also performs routing and NAT.

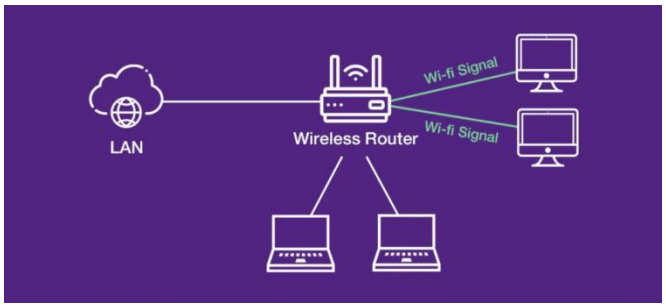

Figure 1.27: *Wireless Access Point (WAP) — provides Wi-Fi connectivity to wireless devices.*

1.6.10 Server

A **server** is a computer or system that provides resources, data, services, or programs to client devices over a network. Servers are central to modern network infrastructure and play a crucial role in web hosting, file sharing, database management, email services, and authentication.

Key characteristics of a Server:

- Provides centralised resources and services to client devices.
- Designed for high availability — typically operates 24 hours a day, 7 days a week.
- Equipped with high-performance hardware: multi-core processors, large RAM, redundant storage (RAID), and multiple NICs.
- Can be physical hardware or a virtual machine running on a hypervisor such as VMware or Hyper-V.

- Communicates with clients using the **client-server model**.

Common types of servers include:

- **Web Server** — hosts websites via HTTP/HTTPS (e.g., Apache, Nginx, IIS).
- **File Server** — stores and manages shared files and folders.
- **Database Server** — hosts and manages databases (e.g., MySQL, PostgreSQL, MS SQL Server).
- **Mail Server** — manages email using SMTP, POP3, and IMAP.
- **DNS Server** — resolves domain names to IP addresses.
- **DHCP Server** — automatically assigns IP addresses to network devices.
- **Authentication Server** — verifies user and device identities (e.g., RADIUS, Active Directory).
- **Proxy Server** — acts as an intermediary providing caching, filtering, and anonymity.

In the context of this project, a dedicated server is used to host the central application, manage user requests, and store data — forming the core of the system's network architecture.

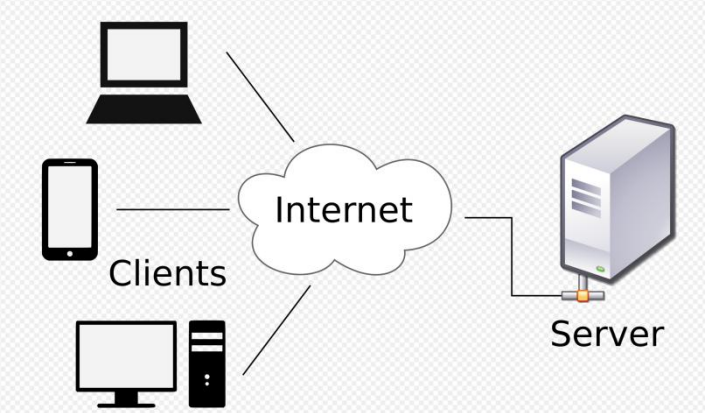

Figure 1.28: *Server — provides centralised resources and services to client devices over the network.*

1.7 Communication Systems

A **communication system** is a collection of individual communication networks, transmission systems, relay stations, tributary stations, and terminal equipment capable of interconnection and interoperation to form an integrated whole. The primary purpose of a communication system is to transfer information from a source to a destination reliably and efficiently.

A basic communication system consists of three fundamental components:

- **Transmitter** — encodes and sends the information signal over the communication medium.
- **Communication Channel** — the physical or wireless medium through which the signal travels (e.g., cable, air, fiber).
- **Receiver** — receives and decodes the signal back into the original information.

Communication systems can be broadly classified based on the transmission medium they use into **wired** and **wireless** systems.

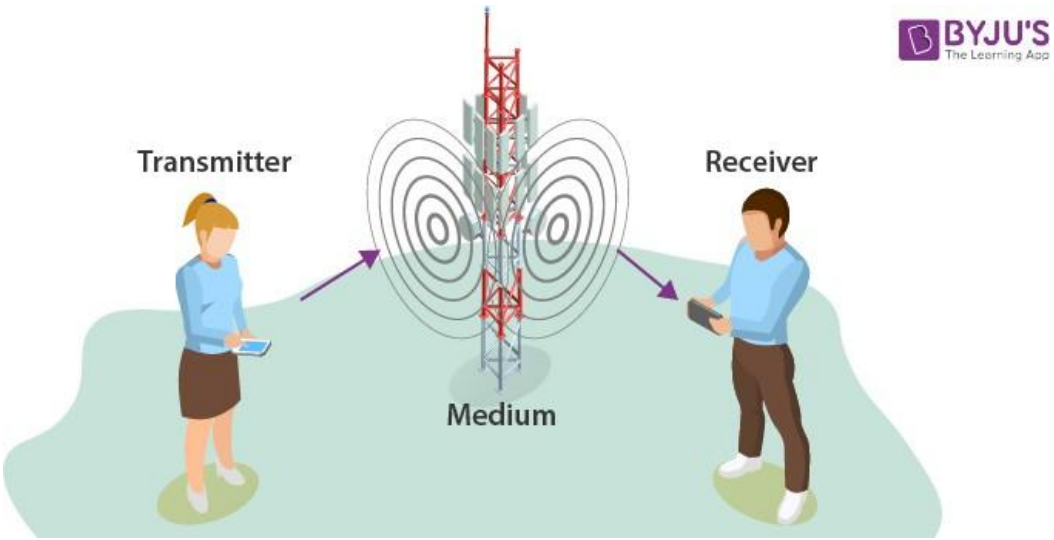

Figure 1.29: *Basic Communication System — transmitter, channel, and receiver.*

1.7.1 Wired versus Wireless Communication

The distinction between wired and wireless communication is one of the most fundamental in networking. Each approach has its own advantages, limitations, and ideal use cases.

Wired Communication

In **wired communication**, data is transmitted through a physical medium such as copper cables, coaxial cables, or fiber-optic cables. The connection is established by a tangible link between the communicating devices.

Advantages of wired communication:

- **High reliability** — less susceptible to interference and signal degradation.
- **High speed** — fiber-optic cables can achieve speeds of hundreds of Gbps.
- **Security** — physical access is required to intercept the signal, making eavesdropping more difficult.
- **Low latency** — consistent and predictable transmission delays.
- **Stable bandwidth** — not affected by the number of nearby users or environmental conditions.

Disadvantages of wired communication:

- **Limited mobility** — devices must remain physically connected to the network.
- **Installation cost** — cabling infrastructure can be expensive and time-consuming to deploy.
- **Scalability constraints** — adding new devices requires additional cabling.

Wireless Communication

In **wireless communication**, data is transmitted over radio waves, microwaves, infrared signals, or satellite links without any physical medium. Devices communicate through the air using standardised protocols such as IEEE 802.11 (Wi-Fi), Bluetooth, Zigbee, and LTE/5G.

Advantages of wireless communication:

- **Mobility** — devices can move freely while remaining connected to the network.
- **Ease of deployment** — no physical cabling required, reducing installation time and cost.

- **Scalability** — new devices can join the network without infrastructure changes.
- **Reach** — can cover areas where cabling is impractical (remote locations, large outdoor spaces).

Disadvantages of wireless communication:

- **Interference** — susceptible to interference from other wireless devices, walls, and environmental obstacles.
- **Security risks** — signals can be intercepted more easily; requires strong encryption (WPA2/WPA3).
- **Variable bandwidth** — performance degrades with distance and number of connected users.
- **Higher latency** — generally higher and less consistent than wired connections.

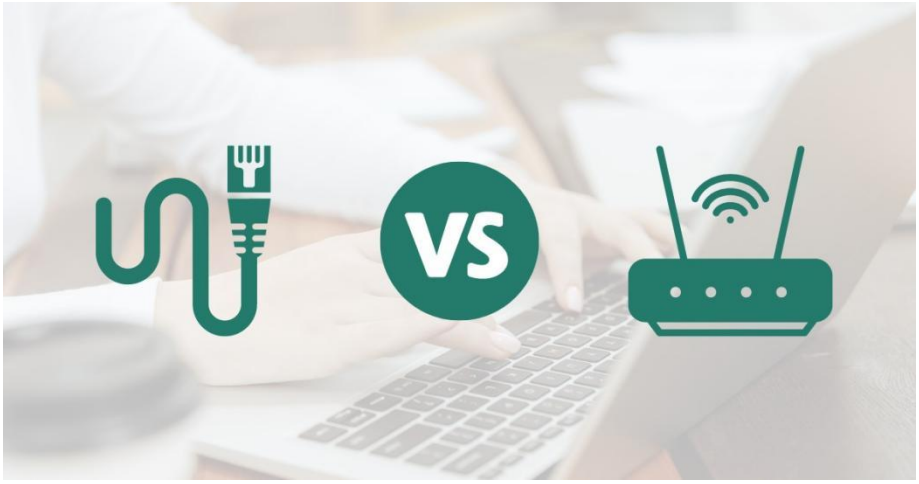

Figure 1.30: *Wired vs Wireless Communication — comparison of the two transmission approaches.*

The table below summarises the key differences between wired and wireless communication:

Table 1.3: *Comparison of Wired and Wireless Communication*

Feature	Wired	Wireless
Medium	Copper, Fiber-optic cable	Radio waves, Microwaves
Speed	Very high (up to 100 Gbps)	Moderate (up to 10 Gbps Wi-Fi 6E)
Mobility	Limited — fixed devices	High — mobile devices
Security	High — physical access needed	Lower — needs encryption
Installation	Complex cabling required	Easy, no cabling needed
Interference	Minimal	High (walls, devices, weather)
Cost	Higher infrastructure cost	Lower deployment cost
Examples	Ethernet, Fiber, DSL	Wi-Fi, Bluetooth, 5G, Zigbee

1.7.2 Wireless Sensor Networks (WSN)

A **Wireless Sensor Network (WSN)** is a distributed network of small, low-power devices called **sensor nodes** (or motes) that are capable of sensing physical or environmental conditions — such as temperature, humidity, pressure, motion, or light — and transmitting the collected data wirelessly to a central point called a **base station** or **sink node**.

WSNs emerged as a natural evolution of wireless communication and are a cornerstone of the **Internet of Things (IoT)**. They enable monitoring and control of physical environments without requiring any fixed infrastructure.

Key characteristics of WSNs:

- **Large-scale deployment** — may consist of hundreds or thousands of sensor nodes deployed over a wide area.
- **Energy constraint** — sensor nodes are typically battery-powered, making energy efficiency

the most critical design consideration.

- **Self-organisation** — nodes can organise themselves into a network without manual configuration.
- **Multi-hop communication** — data is relayed from node to node until it reaches the sink, extending the network range.
- **Limited resources** — sensor nodes have constrained processing power, memory, and bandwidth.
- **Fault tolerance** — the network continues to function even if individual nodes fail.

A typical WSN node consists of four components:

- **Sensing Unit** — one or more sensors that measure physical phenomena and convert them into electrical signals.
- **Processing Unit** — a microcontroller that processes the sensed data and controls the node's operation.
- **Communication Unit** — a radio transceiver that transmits and receives data wirelessly.
- **Power Unit** — a battery or energy harvesting module (e.g., solar cell) that powers the node.

Common applications of WSNs include:

- **Environmental monitoring** — forest fire detection, flood monitoring, and air quality measurement.
- **Healthcare** — patient monitoring, wearable health sensors, and hospital asset tracking.
- **Smart agriculture** — soil moisture monitoring, crop health tracking, and automated irrigation.
- **Industrial automation** — machine condition monitoring, predictive maintenance, and factory automation.
- **Smart cities** — traffic monitoring, smart street lighting, and waste management.
- **Military and security** — battlefield surveillance, intrusion detection, and border monitoring.

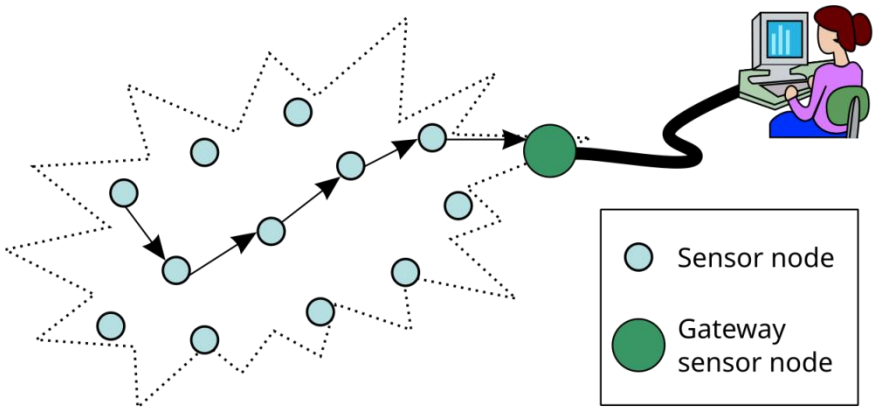

Figure 1.31: *Wireless Sensor Network (WSN) — distributed sensor nodes collecting data and relaying it to a base station.*

1.7.3 Ad Hoc Networks

An **Ad Hoc network** is a type of wireless network that is formed **spontaneously and temporarily** without relying on any pre-existing fixed infrastructure such as access points, routers, or base stations. Instead, each device (node) in the network communicates directly with other nodes and can also act as a **router**, forwarding data on behalf of other nodes.

The term *ad hoc* is Latin for "for this purpose", reflecting the on-demand, temporary nature of these networks. Ad hoc networks are closely related to WSNs and are the underlying architecture for many modern wireless technologies including Bluetooth mesh, Zigbee, and mobile peer-to-peer networks.

Key characteristics of Ad Hoc networks:

- **Infrastructure-less** — no fixed access points or routers; all nodes are equal peers.
- **Self-configuring** — nodes automatically discover each other and configure routing without manual intervention.
- **Dynamic topology** — the network topology changes as nodes move, join, or leave the network.
- **Distributed operation** — no central controller; each node participates in routing and forwarding decisions.
- **Multi-hop routing** — data may pass through multiple intermediate nodes to reach its destination.
-

- **Limited resources** — nodes typically have constrained battery life, processing power, and bandwidth.

There are two major types of Ad Hoc networks:

- **MANET (Mobile Ad Hoc Network)** — a network of mobile devices (laptops, smartphones) that communicate directly with each other. MANETs are used in disaster recovery, military operations, and temporary events where infrastructure is unavailable.
- **VANET (Vehicular Ad Hoc Network)** — a special form of MANET designed for vehicles. Cars communicate with each other (V2V) and with roadside infrastructure (V2I) to enable applications such as collision avoidance, traffic management, and autonomous driving support.

Comparison between Infrastructure-based and Ad Hoc networks:

Table 1.4: *Infrastructure-based vs Ad Hoc Networks*

Feature	**Infrastructure-based**	**Ad Hoc**
Infrastructure	Requires fixed AP/router	No fixed infrastructure
Setup	Pre-configured	Self-configuring
Topology	Fixed (star)	Dynamic, changes over time
Scalability	High	Limited
Range	Extended via APs	Limited by node proximity
Use cases	Offices, homes, campuses	Military, disaster, WSN

Ad Hoc networks play a critical role in scenarios where deploying traditional infrastructure is impractical, such as disaster relief operations, battlefield communications, and remote sensor deployments. They also form the foundation of WSNs, as sensor nodes typically communicate in an ad hoc fashion to relay data back to the base station.

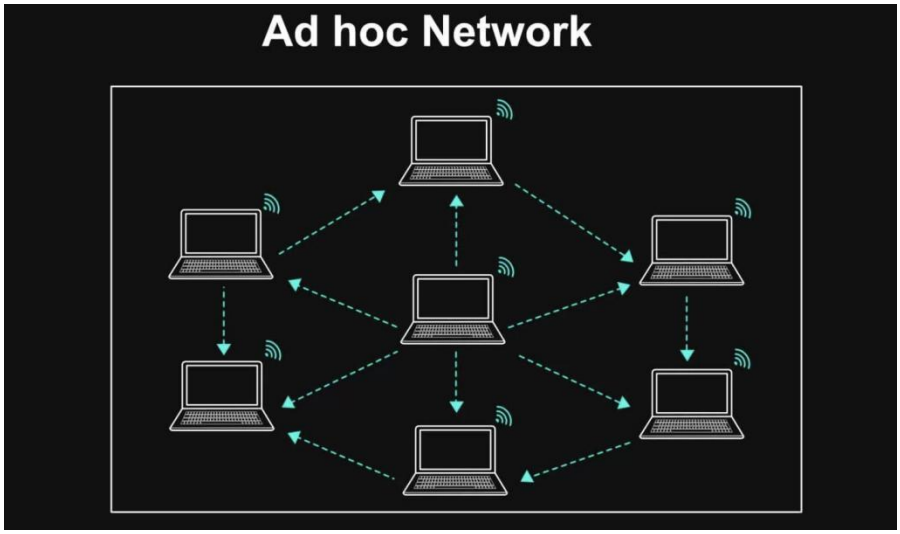

Figure 1.32: *Ad Hoc Network — nodes communicating directly without any fixed infrastructure.*

Chapter 2

Wireless Sensor Network (WSN)

2.1 Definition of a Wireless Sensor Network

A **Wireless Sensor Network (WSN)** is a distributed network composed of a large number of small, autonomous devices called **sensor nodes** (also known as *motes*), which are spatially deployed across a geographic area to monitor and collect data about physical or environmental conditions. These conditions include temperature, humidity, pressure, light intensity, sound, vibration, motion, and many other parameters, depending on the application domain.

Each sensor node is equipped with one or more sensing units, a small microcontroller for local data processing, a wireless transceiver for communication, and a limited power source — typically a battery. The nodes communicate wirelessly with one another and relay the collected data, either directly or through intermediate nodes, to a central point known as the **base station** or **sink node**. From there, the data is forwarded to end users or integrated into a broader information system for further analysis and decision-making.

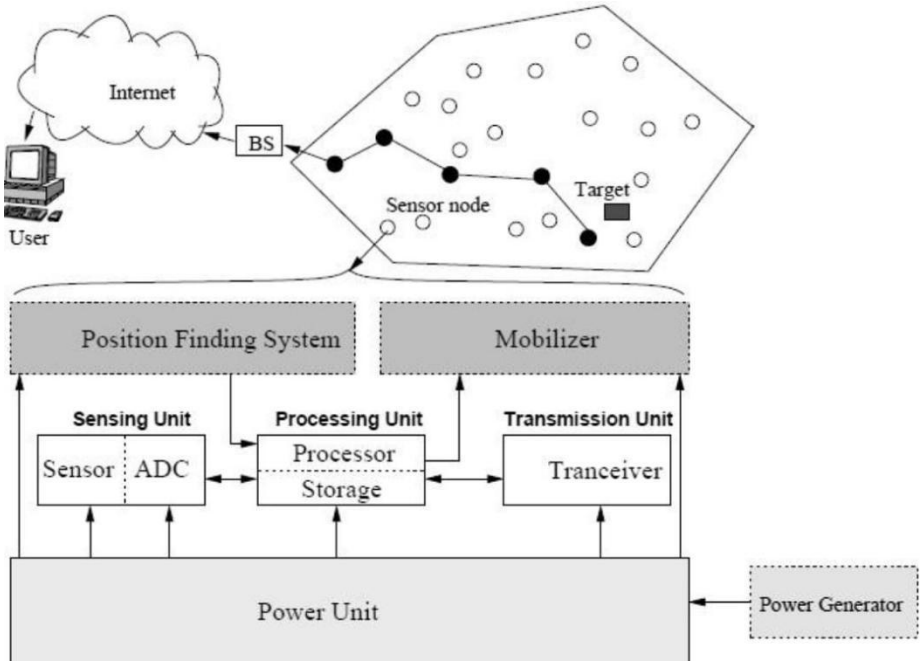

Figure 2.33: *General overview of a Wireless Sensor Network (WSN) — sensor nodes deployed across an area, communicating wirelessly with a base station.*

WSNs are fundamentally characterised by their ability to operate in **unattended** and often **harsh environments**, where deploying traditional wired infrastructure would be impractical or impossible. This makes them particularly valuable in remote monitoring scenarios such as disaster zones, forests, battlefields, and industrial facilities. The wireless and self-organising nature of WSNs allows them to be rapidly deployed with minimal human intervention.

A key distinction of WSNs compared to conventional wireless networks lies in their design priorities. Rather than maximising data throughput or communication range, WSNs are optimised for:

- **Energy efficiency** — since sensor nodes rely on batteries that are difficult or impossible to recharge or replace once deployed, minimising power consumption is the foremost design constraint.
- **Scalability** — a WSN may consist of tens to thousands of nodes; the network must function correctly and efficiently regardless of scale.
- **Fault tolerance** — individual nodes may fail due to energy depletion, physical damage, or environmental conditions; the network must continue to operate reliably despite such failures.
- **Self-organisation** — nodes must autonomously configure themselves, discover neighbours, and establish communication routes without centralised control or manual configuration.
- **Low cost** — each individual sensor node must be inexpensive to manufacture and deploy, enabling large-scale deployment across wide areas.

WSNs emerged from military applications in the 1980s and 1990s — most notably the **DARPA SensIT programme** — before expanding into civilian domains. Today, they form a foundational technology in the **Internet of Things (IoT)** ecosystem, enabling smart environments in agriculture, healthcare, industrial automation, smart cities, and environmental monitoring.

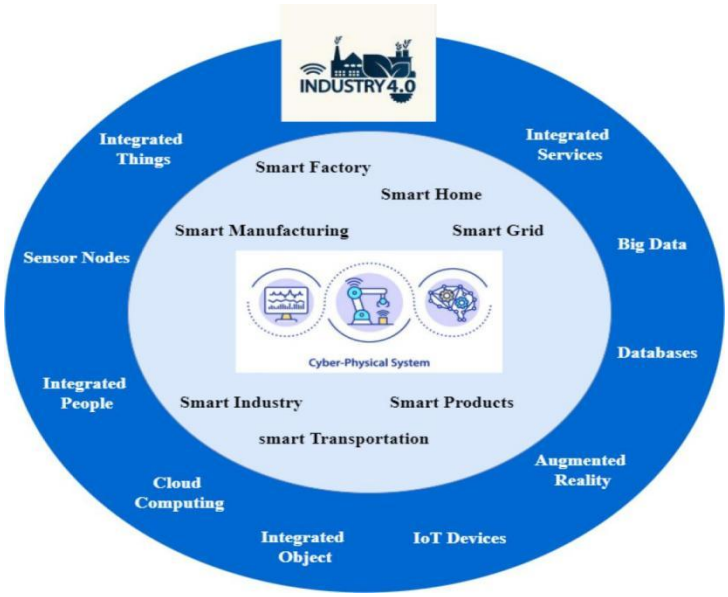

Figure 2.34: *Application domains of Wireless Sensor Networks — from environmental monitoring and precision agriculture to healthcare and smart cities.*

2.2 Importance and Applications of Wireless Sensor Networks

The rapid advancement of microelectronics, wireless communication, and embedded systems has elevated WSNs from a niche military technology into one of the most impactful and versatile sensing paradigms of the modern era. Their ability to collect real-time data from physically distributed locations — without the need for wired infrastructure — makes them indispensable across a wide spectrum of domains. The importance of WSNs stems not only from what they sense, but from *how* they do it: autonomously, continuously, and at scale.

2.2.1 Importance of Wireless Sensor Networks

WSNs address fundamental challenges that traditional monitoring systems cannot solve efficiently. Their significance can be understood through the following dimensions:

- **Real-time monitoring and situational awareness** — WSNs provide continuous, live data streams from the field, enabling operators and automated systems to make timely and informed decisions. This is critical in time-sensitive scenarios such as fire detection, flood warning, and patient health monitoring.
- **Deployment in inaccessible or hazardous environments** — sensor nodes can be placed in locations where human presence is dangerous or physically impossible, such as volcanic regions, nuclear facilities, deep-sea environments, or conflict zones, providing invaluable data without risk to human life.
- **Infrastructure-free operation** — unlike wired networks, WSNs require no pre-existing physical infrastructure. This drastically reduces deployment time and cost, making them ideal for temporary installations, emergency response, and remote rural areas where cabling is infeasible.
- **Scalability and dense coverage** — a WSN can be scaled from a few dozen nodes to tens of thousands, enabling fine-grained spatial coverage over large geographic areas. This density of sensing is unachievable with traditional monitoring stations.
- **Integration with IoT and data-driven systems** — WSNs serve as the physical sensing layer of the Internet of Things (IoT), feeding data into cloud platforms, machine learning pipelines, and decision-support systems, enabling smart and adaptive environments.

- **Cost-effectiveness** — the low unit cost of sensor nodes, combined with the elimination of wiring infrastructure, results in a highly cost-effective solution for large-scale monitoring compared to traditional wired sensor installations.

2.2.2 Applications of Wireless Sensor Networks

The versatility of WSNs has led to their adoption across a broad range of industries and research fields. The following subsections highlight the most prominent application domains.

Environmental Monitoring

One of the earliest and most widespread applications of WSNs is environmental monitoring. Sensor nodes are deployed across forests, rivers, coastlines, and atmospheric zones to continuously measure parameters such as temperature, humidity, soil moisture, air quality, and pollutant levels.

Key use cases include:

- **Forest fire detection** — densely deployed nodes detect sudden rises in temperature and smoke concentration, triggering early warnings before fires spread uncontrollably.
- **Flood and earthquake monitoring** — water-level sensors and seismic nodes provide early-warning systems that can save lives in disaster-prone regions.
- **Air and water quality monitoring** — nodes deployed in urban areas or near industrial sites measure pollution levels in real time, supporting environmental regulation and public health.
- **Wildlife habitat tracking** — miniaturised sensor nodes attached to animals or placed in ecosystems collect behavioural and environmental data without disturbing natural habitats.

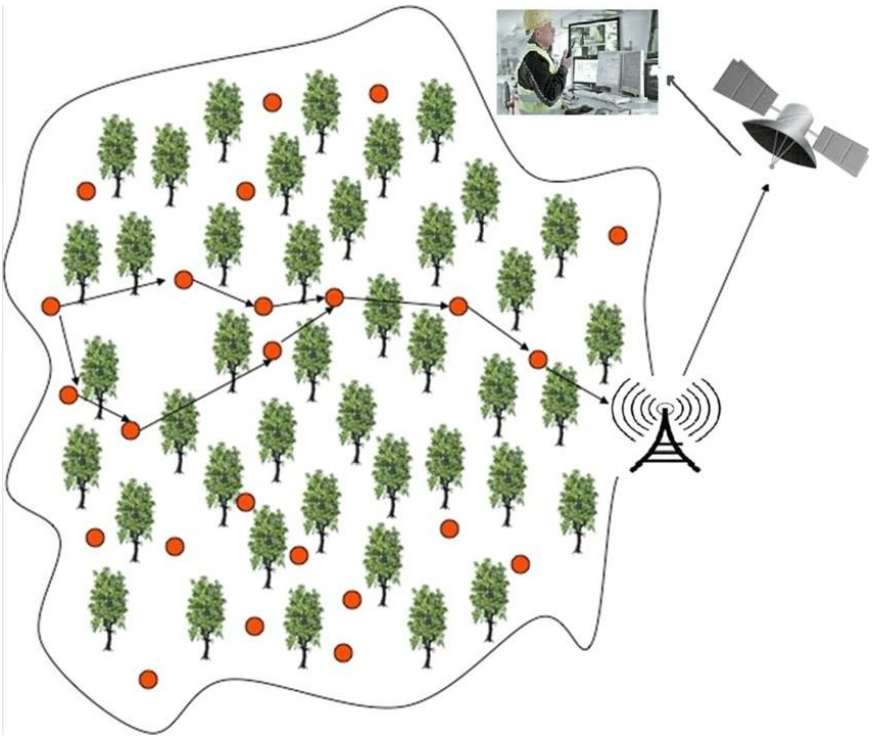

Figure 2.35: *WSN for environmental monitoring — nodes deployed across a forest to detect fire, measure humidity, and monitor air quality.*

Healthcare and Medical Monitoring

WSNs have transformed the field of healthcare by enabling continuous, non-invasive patient monitoring both inside and outside clinical settings. Wearable or implantable sensor nodes measure vital signs such as heart rate, blood pressure, body temperature, blood oxygen saturation, and glucose levels, transmitting data wirelessly to healthcare providers in real time.

Key use cases include:

- **Remote patient monitoring** — elderly or chronically ill patients can be monitored from home, reducing hospital admissions and enabling early intervention when anomalies are detected.
- **Smart hospital systems** — WSNs track patient location, monitor equipment usage, and manage environmental conditions (temperature, humidity) in wards and operating theatres.
- **Post-operative recovery tracking** — implanted or wearable sensors transmit recovery metrics to physicians, enabling personalised and timely post-surgical care.
- **Emergency response** — first responders equipped with body-area sensor networks can have their physiological status monitored in real time during hazardous operations.

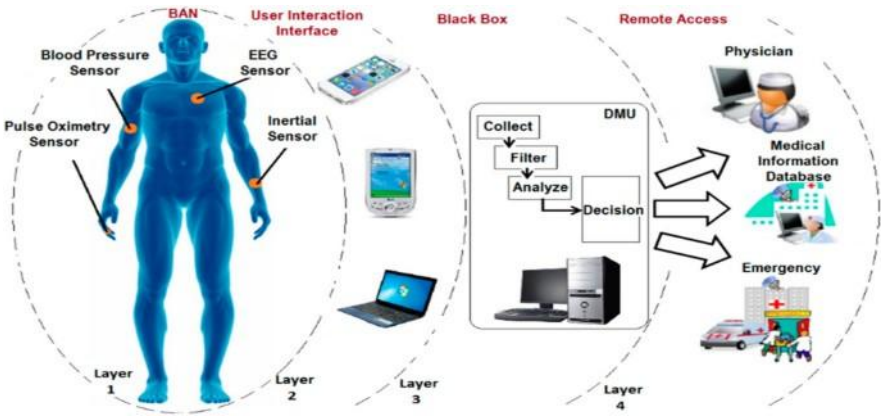

Figure 2.36: *WSN in healthcare — wearable sensor nodes monitoring patient vital signs and transmitting data to a medical base station.*

Precision Agriculture

Agriculture is one of the most rapidly growing domains for WSN deployment. Sensor nodes distributed across farmland measure soil moisture, nutrient levels, temperature, and sunlight exposure, enabling **precision agriculture** — the practice of applying the right resources (water, fertiliser, pesticides) at the right place and time, maximising yield while minimising waste.

Key use cases include:

- **Automated irrigation systems** — soil moisture sensors trigger irrigation only when and where water is needed, conserving water resources significantly.
- **Crop health monitoring** — sensors detect early signs of disease, pest infestation, or nutrient deficiency, allowing targeted intervention before crop loss occurs.
- **Greenhouse management** — WSNs regulate temperature, humidity, and CO_2 levels inside greenhouses automatically to optimise growing conditions.
- **Livestock monitoring** — sensors attached to animals track location, activity, and health metrics, improving herd management and early detection of illness.

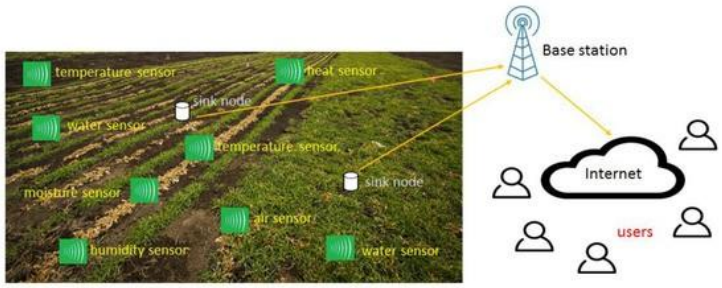

Figure 2.37: *WSN in precision agriculture — soil and climate sensor nodes transmitting data to optimise irrigation and crop management.*

Industrial Monitoring and Automation

In industrial environments, WSNs are deployed to monitor machinery, production processes, and facility conditions. This application domain is commonly referred to as the **Industrial Internet of Things (IIoT)**.

Key use cases include:

- **Predictive maintenance** — vibration, temperature, and acoustic sensors on industrial machinery detect early signs of mechanical wear or failure, allowing maintenance to be scheduled before a breakdown occurs, reducing downtime and repair costs.
- **Structural health monitoring** — sensor nodes embedded in bridges, dams, pipelines, and buildings continuously monitor stress, strain, and deformation, ensuring structural integrity over time.
- **Pipeline and utility monitoring** — WSNs detect leaks, pressure anomalies, and corrosion in oil, gas, and water pipelines across remote or hazardous terrain.
- **Factory floor automation** — sensor networks provide real-time feedback to control systems, enabling adaptive and efficient manufacturing processes.

Figure 2.38: *WSN in industrial monitoring — sensor nodes deployed on machinery and infrastructure for predictive maintenance and process control.*

Military and Security Applications

WSNs were originally developed for military use and remain a critical technology in defence and security applications. Their ability to be covertly deployed and to operate autonomously in hostile environments makes them uniquely suited for battlefield and border surveillance.

Key use cases include:

- **Battlefield surveillance** — sensor nodes detect enemy movement, vehicle presence, and acoustic signatures across a theatre of operations, providing commanders with real-time situational awareness.
- **Border monitoring** — WSNs deployed along national borders detect unauthorised crossings through motion, heat, and acoustic sensors, alerting security forces without requiring continuous human patrol.
- **Target tracking** — distributed sensor arrays track the movement of personnel or vehicles across a monitored zone.
- **Nuclear, biological, and chemical (NBC) detection** — specialised sensor nodes detect hazardous agents in the environment, providing early warning of NBC threats.

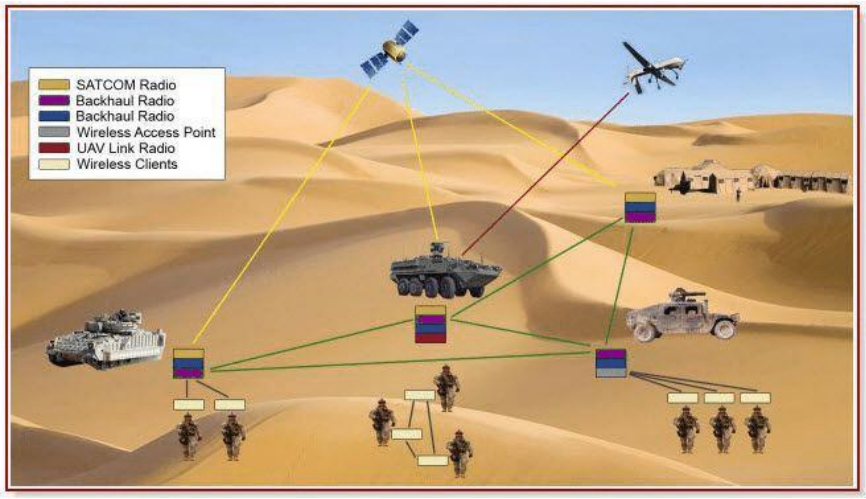

Figure 2.39: *WSN in military applications — covertly deployed sensor nodes providing battlefield surveillance and perimeter monitoring.*

Smart Cities and Infrastructure

WSNs are a cornerstone technology in the development of **smart cities**, where urban infrastructure is instrumented and interconnected to improve efficiency, sustainability, and quality of life for citizens.

Key use cases include:

- **Smart traffic management** — road-embedded and roadside sensors monitor vehicle flow and density, feeding data to adaptive traffic signal controllers that reduce congestion and emissions.
- **Smart street lighting** — light and motion sensors adjust street lighting intensity based on ambient conditions and pedestrian presence, reducing energy consumption.
- **Waste management** — sensors in rubbish bins report fill levels to centralised systems, enabling optimised and on-demand collection routes.
- **Smart parking** — ground-embedded sensors detect vehicle occupancy in parking spaces and relay availability information to drivers via mobile applications.
- **Public safety and surveillance** — acoustic and environmental sensors detect anomalies such as gunshots, gas leaks, or extreme weather events, enabling rapid emergency response.

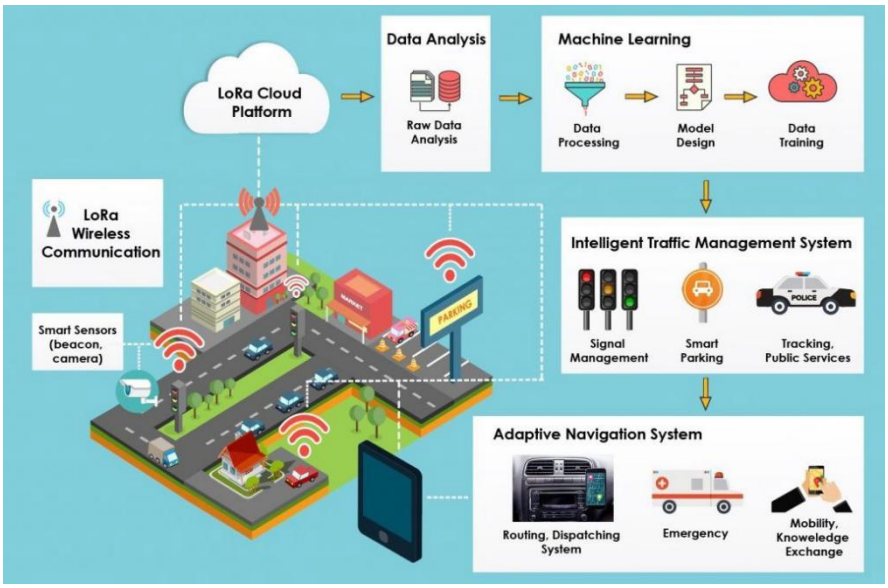

Figure 2.40: *WSN in smart city infrastructure — sensor nodes integrated into urban systems for traffic management, lighting, and public safety.*

2.3 Basic Components of a Wireless Sensor Network

A Wireless Sensor Network is composed of several key components that work together to enable sensing, processing, and communication. Understanding these components is essential to appreciating how a WSN functions as a complete system. The main components of a WSN are: the **sensor node**, the **base station**, the **communication network**, and the **task manager** (end user or application layer).

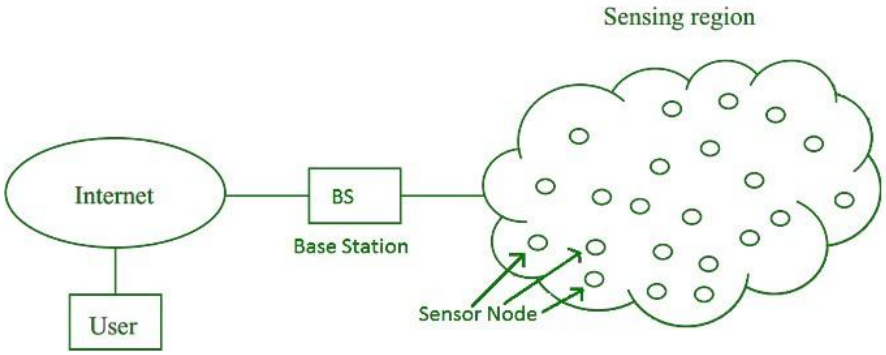

Figure 2.41: *Basic components of a Wireless Sensor Network — sensor nodes, base station, communication links, and end-user application.*

2.3.1 Sensor Node

The **sensor node** is the fundamental building block of any WSN. It is a small, embedded device responsible for sensing physical phenomena in its surrounding environment, processing the collected data locally, and transmitting it wirelessly to neighbouring nodes or directly to the base station. Despite their small size and low cost, sensor nodes are complete embedded systems capable of autonomous operation.

A typical sensor node consists of four core subsystems:

Sensing Unit

The sensing unit is responsible for detecting and measuring physical or environmental parameters such as temperature, humidity, pressure, light intensity, sound, vibration, or chemical concentration. It consists of two components:

- **Sensor** — a transducer that converts a physical quantity into an analogue electrical signal. Common sensor types include thermistors (temperature), photodiodes (light), piezoelectric elements (vibration), and electrochemical sensors (gas or pH).

- **Analogue-to-Digital Converter (ADC)** — converts the continuous analogue signal produced by the sensor into a discrete digital value that can be processed by the microcontroller.

Processing Unit

The processing unit is the computational core of the sensor node. It consists of:

- **Microcontroller (MCU)** — executes the node's embedded software, controls all subsystems, performs local data processing (filtering, aggregation, compression), and manages the communication and power subsystems. Low-power microcontrollers such as the ATmega328 (Arduino) and MSP430 (Texas Instruments) are commonly used in WSN applications.
- **Memory** — stores the operating firmware, sensor readings, and intermediate processing results. Typically includes Flash memory (program storage) and SRAM (runtime data storage).

Communication Unit

The communication unit enables wireless data exchange between sensor nodes and with the base station. It consists of:

- **Transceiver** — a combined transmitter and receiver that sends and receives radio frequency (RF) signals. The transceiver is typically the most power-hungry component of the node and is designed to support multiple operating modes (transmit, receive, idle, sleep) to conserve energy.
- **Antenna** — radiates and receives electromagnetic signals. Most sensor nodes use compact printed or chip antennas operating in the ISM bands (433 MHz, 868 MHz, 2.4 GHz).
- Common wireless standards used in WSNs include **IEEE 802.15.4** (ZigBee), **Bluetooth Low Energy (BLE)**, **LoRa**, and **Z-Wave**, each offering different trade-offs between range, data rate, and power consumption.

Power Unit

The power unit supplies energy to all subsystems of the sensor node. It is the most critical constraint in WSN design, as node lifetime is directly determined by energy availability.

- **Battery** — the primary power source for most sensor nodes. Alkaline, lithium, or rechargeable batteries are used depending on the application. Battery capacity is limited, making energy efficiency a paramount design concern.
- **Energy harvesting** — some advanced nodes supplement or replace batteries with energy harvested from the environment, such as solar panels (photovoltaic cells), thermoelectric generators (heat gradients), piezoelectric harvesters (vibration), or RF energy harvesting. This can significantly extend node lifetime or enable perpetual operation.
- **Power management circuit** — regulates voltage levels, manages sleep/wake cycles, and ensures efficient energy distribution across subsystems.

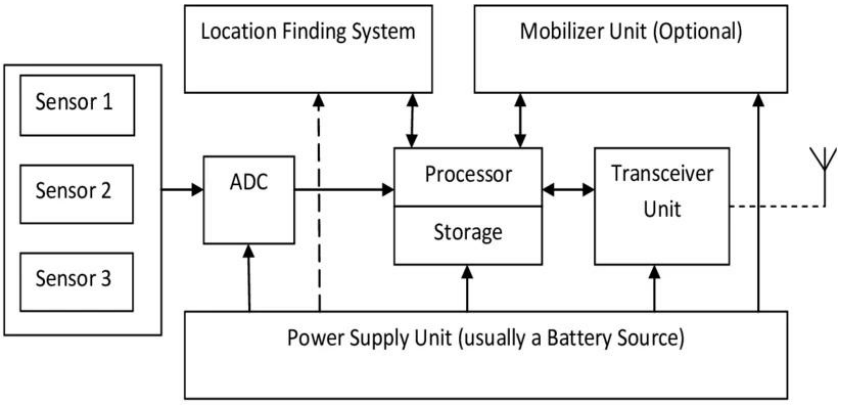

Figure 2.42: *Internal architecture of a WSN sensor node — sensing, processing, communication, and power subsystems.*

2.3.2 Base Station (Sink Node)

The **base station**, also referred to as the **sink node** or **gateway**, is the central collection and coordination point of the WSN. Unlike ordinary sensor nodes, the base station is typically not constrained by energy or processing limitations, as it is often connected to a continuous power supply and a more powerful computing platform.

Key roles of the base station include:

- **Data aggregation** — collects data packets forwarded by sensor nodes from across the network and aggregates them into a coherent dataset for further processing.
- **Protocol gateway** — bridges the WSN's low-power wireless protocol (e.g., ZigBee, LoRa) with standard networking protocols (e.g., TCP/IP, HTTP) to enable connectivity to the internet or a local area network.
- **Network management** — may issue configuration commands, software updates, or query requests to nodes in the field.
- **Data forwarding** — transmits the aggregated sensor data to a remote server, cloud platform, or end-user application for storage, visualisation, and analysis.

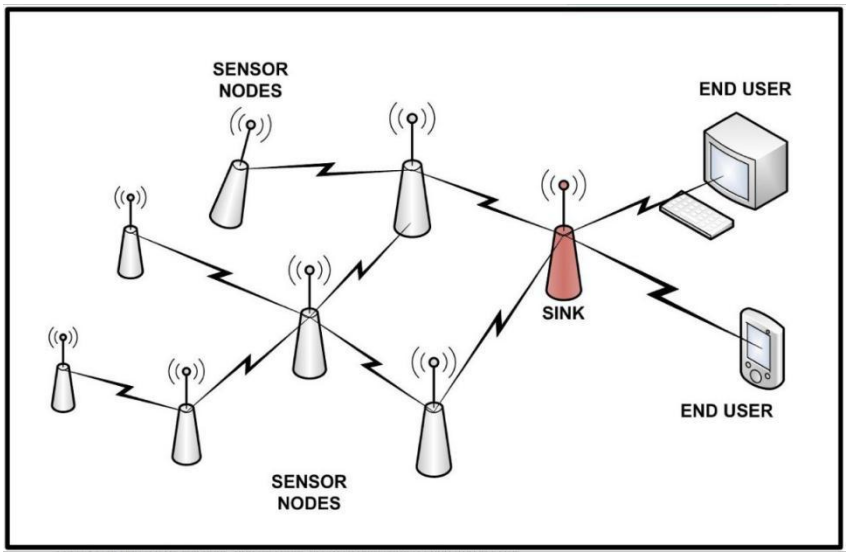

Figure 2.43: *WSN base station (sink node) — collecting data from sensor nodes and forwarding it to an external network or application server.*

2.3.3 Communication Network

The **communication network** defines the wireless links and routing structure through which sensor nodes exchange data with one another and with the base station. It encompasses both the physical transmission medium and the networking protocols that govern how data is routed across the WSN.

- **Single-hop communication** — sensor nodes transmit their data directly to the base station in a single wireless hop. Simple and energy-efficient for small, dense networks where all nodes are within range of the base station.
- **Multi-hop communication** — data is relayed from node to node across multiple hops before reaching the base station. This extends the effective range of the network and is essential in large-scale deployments where nodes may be far from the sink.
- **Routing protocols** — specialised energy-aware routing protocols such as **LEACH** (Low-Energy Adaptive Clustering Hierarchy), **PEGASIS**, and **SPIN** are used to determine efficient data paths that minimise energy consumption and prolong network lifetime.

2.3.4 Task Manager (End-User Application)

The **task manager** represents the end-user side of the WSN — the software application, platform, or human operator that defines the network's objectives, issues queries or tasks to the sensor nodes, and consumes the data produced by the network.

- **Application software** — processes, visualises, and stores the sensor data received from the base station. Examples include SCADA systems, IoT cloud dashboards (e.g., ThingSpeak, AWS IoT), and custom monitoring applications.
- **Task dissemination** — the task manager may push instructions back into the network via the base station, such as adjusting sensing frequency, activating specific nodes, or reconfiguring routing parameters.
- **Decision-making** — in automated systems, the application layer triggers actuators or alerts based on sensor data thresholds, closing the loop between sensing and response (e.g., activating irrigation pumps, sounding alarms, or notifying emergency services).

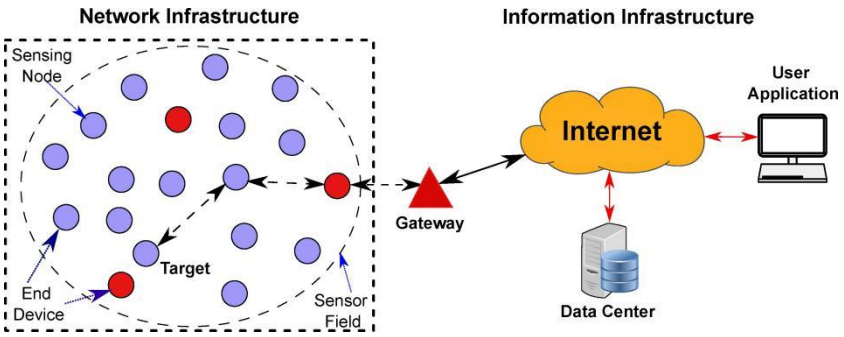

Figure 2.44: *WSN task manager — end-user application receiving, visualising, and acting upon data collected from the sensor network.*

2.4 WSN Network Architecture

The architecture of a Wireless Sensor Network defines how sensor nodes are organised, how they communicate with one another, and how data ultimately reaches the base station. The choice of architecture has a direct impact on the network's energy efficiency, scalability, latency, and overall lifetime. WSN architectures are generally classified into three principal categories: **flat (single-tier) architecture**, **hierarchical (clustered) architecture**, and **hybrid architecture**.

2.4.1 Flat (Single-Tier) Architecture

In a **flat architecture**, all sensor nodes are functionally identical and play the same role within the network. There is no predefined hierarchy: every node senses data, performs minimal local processing, and forwards its data toward the base station using **multi-hop routing**, relaying packets through neighbouring nodes until they reach the sink.

Characteristics:

- All nodes have equal responsibilities and capabilities.
- Routing decisions are typically made collaboratively, often using protocols such as flooding, gossiping, or data-centric routing (e.g., directed diffusion).
- Simple to deploy, since no role assignment or cluster formation is required.
- Suffers from poor scalability in very large networks, as nodes near the base station experience disproportionately high traffic loads and deplete their energy faster than nodes farther away — a phenomenon known as the **energy hole** or **hot spot** problem.

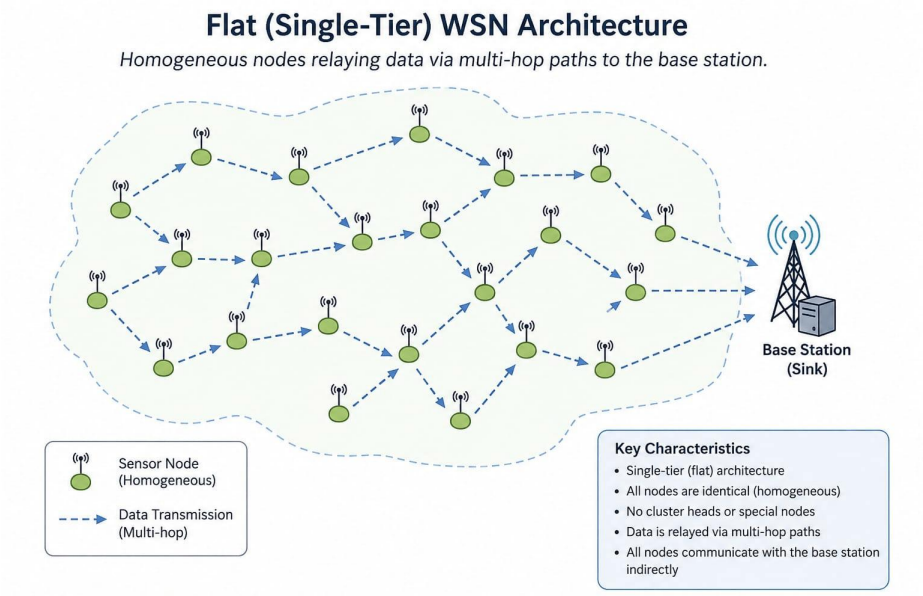

Figure 2.45: *Flat (single-tier) WSN architecture — homogeneous nodes relaying data via multi-hop paths to the base station.*

2.4.2 Hierarchical (Clustered) Architecture

In a **hierarchical architecture**, nodes are organised into groups called **clusters**. Within each cluster, one node is elected or designated as the **cluster head (CH)**, while the remaining nodes act as **cluster members**. Cluster members sense data and transmit it to their respective cluster head, which aggregates, compresses, and forwards the combined data — either directly or through other cluster heads — to the base station.

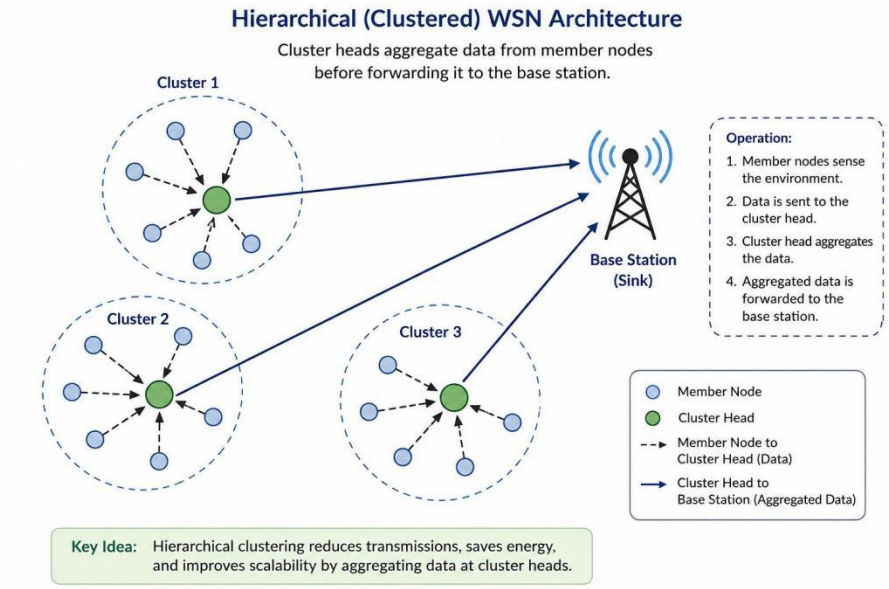
Hierarchical (Clustered) WSN Architecture
Cluster heads aggregate data from member nodes
before forwarding it to the base station.
Cluster 1
Cluster 2
Cluster 3
Base Station
(Sink)
Operation:
1. Member nodes sense the environment.
2. Data is sent to the cluster head.
3. Cluster head aggregates the data.
4. Aggregated data is forwarded to the base station.
Member Node
Cluster Head
Member Node to Cluster Head (Data)
Cluster Head to Base Station (Aggregated Data)
Key Idea: Hierarchical clustering reduces transmissions, saves energy, and improves scalability by aggregating data at cluster heads.

Figure 2.46: *Hierarchical (clustered) WSN architecture — cluster heads aggregating data from member nodes before forwarding it to the base station.*

Characteristics:

- Reduces the volume of data transmitted over long distances through local data aggregation at the cluster head, significantly lowering overall energy consumption.
- Improves scalability, since the network is logically divided into manageable sub-groups.
- Cluster heads incur a higher energy burden than ordinary nodes; protocols such as **LEACH** (Low-Energy Adaptive Clustering Hierarchy) address this by periodically rotating the cluster-head role among nodes to balance energy consumption across the network.
- Introduces additional complexity in cluster formation, cluster-head election, and inter-cluster communication management.

2.4.3 Hybrid Architecture

A **hybrid architecture** combines elements of both flat and hierarchical designs, often introducing heterogeneous node types with differing capabilities. For example, a network may include ordinary low-power sensor nodes alongside a smaller number of more capable **relay nodes** or **gateway nodes** that handle long-range communication and data aggregation, while the majority of nodes communicate using flat, peer-to-peer routing within local neighbourhoods.

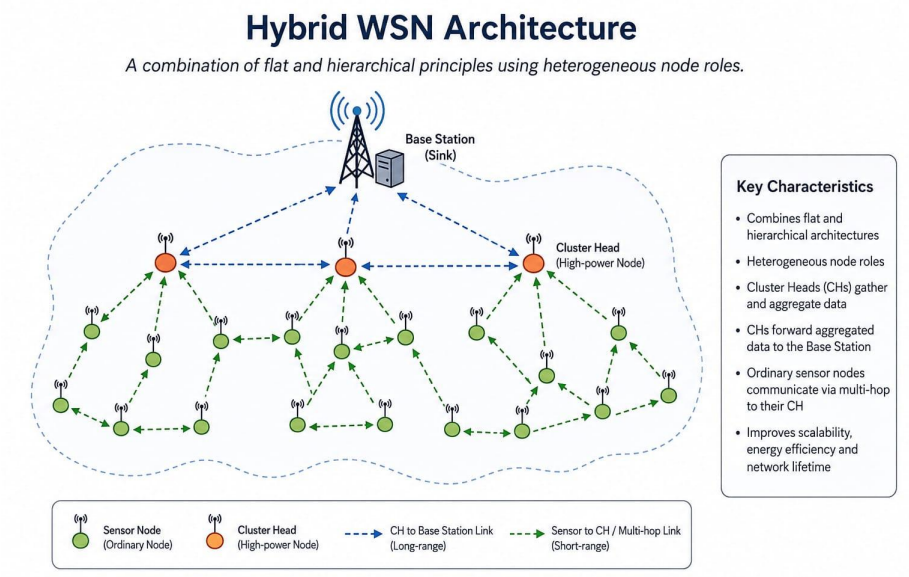
Hybrid WSN Architecture
A combination of flat and hierarchical principles using heterogeneous node roles.
Base Station
(Sink)
Cluster Head
(High-power Node)
Key Characteristics
Combines flat and hierarchical architectures
Heterogeneous node roles
Cluster Heads (CHs) gather and aggregate data
CHs forward aggregated data to the Base Station
Ordinary sensor nodes communicate via multi-hop to their CH
Improves scalability, energy efficiency and network lifetime
Sensor Node
(Ordinary Node)
Cluster Head
(High-power Node)
CH to Base Station Link
(Long-range)
Sensor to CH / Multi-hop Link
(Short-range)

Figure 2.47: *Hybrid WSN architecture — a combination of flat and hierarchical principles using heterogeneous node roles.*

Characteristics:

- Leverages the simplicity of flat architectures at the local level while benefiting from the energy savings and scalability of hierarchical aggregation at higher levels.
- Well-suited to large, heterogeneous deployments where some nodes can be equipped with greater processing power, memory, or energy reserves than others.
- Offers a flexible trade-off between deployment simplicity and long-term network efficiency, making it common in modern, real-world WSN and IoT deployments.

2.5 Data Flow in WSN

Data flow in a Wireless Sensor Network describes the journey of information from the moment it is sensed at a node to the moment it becomes usable knowledge for an end user or application. This flow generally proceeds through four sequential stages: **sensing**, **processing**, **communication/routing**, and **data aggregation at the base station**.

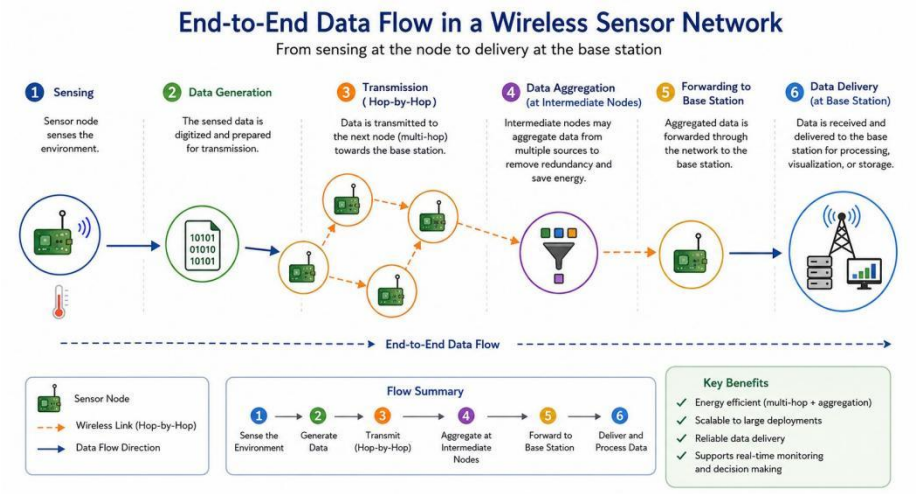
End-to-End Data Flow in a Wireless Sensor Network
From sensing at the node to delivery at the base station
1 Sensing
Sensor node senses the environment.
2 Data Generation
The sensed data is digitized and prepared for transmission.
3 Transmission (Hop-by-Hop)
Data is transmitted to the next node (multi-hop) towards the base station.
4 Data Aggregation (at Intermediate Nodes)
Intermediate nodes may aggregate data from multiple sources to remove redundancy and save energy.
5 Forwarding to Base Station
Aggregated data is forwarded through the network to the base station.
6 Data Delivery (at Base Station)
Data is received and delivered to the base station for processing, visualization, or storage.
10101
01010
10101
End-to-End Data Flow
Sensor Node
Wireless Link (Hop-by-Hop)
Data Flow Direction
Flow Summary
Sense the Environment
Generate Data
Transmit (Hop-by-Hop)
Aggregate at Intermediate Nodes
Forward to Base Station
Deliver and Process Data
Key Benefits
Energy efficient (multi-hop + aggregation)
Scalable to large deployments
Reliable data delivery
Supports real-time monitoring and decision making

Figure 2.48: *End-to-end data flow in a Wireless Sensor Network, from sensing at the node to delivery at the base station.*

2.5.1 Sensing

The process begins at the sensing unit of each node, where physical phenomena — temperature, humidity, vibration, light, and so on — are captured and converted into digital readings via an analogue-to-digital converter. The sampling rate and sensing schedule are typically configured according to the application's requirements, balancing data resolution against energy consumption.

2.5.2 Local Processing

Once a reading is obtained, the node's processing unit may perform local operations before transmission. These operations can include filtering noise, compressing data, detecting threshold crossings or anomalies, and, in some architectures, partially aggregating data received from neighbouring nodes. Performing computation locally rather than transmitting raw data is a key strategy for conserving energy, since wireless transmission is typically far more energy-intensive than local processing.

2.5.3 Communication and Routing

Processed data is then transmitted wirelessly toward the base station. Depending on the network architecture, this may occur through:

- **Direct transmission** — where a node sends data straight to the base station, suitable only for small networks or nodes located close to the sink.
- **Multi-hop routing** — where data is relayed through a sequence of intermediate nodes, each forwarding the packet closer to its destination, common in flat architectures.
- **Cluster-based forwarding** — where data first travels to a cluster head for aggregation before being forwarded onward, common in hierarchical architectures.

Routing protocols are responsible for determining the most efficient path for data to travel, taking into account factors such as remaining node energy, link quality, and distance to the base station.

2.5.4 Data Aggregation and Delivery

As data converges toward the base station, it is often aggregated — combined with data from multiple nodes to eliminate redundancy and reduce the overall volume of transmitted information. The base station then forwards the consolidated data to external systems, where it may be stored, visualised, or fed into analytical and decision-support applications.

2.6 WSN Protocol Stack

Communication within a Wireless Sensor Network is governed by a layered **protocol stack**, conceptually similar to the OSI model but adapted to the specific constraints of sensor networks, such as limited energy, memory, and processing power. The WSN protocol stack typically comprises five layers: **physical**, **data link**, **network**, **transport**, and **application**, often complemented by three cross-layer planes that manage power, mobility, and tasks across all layers.

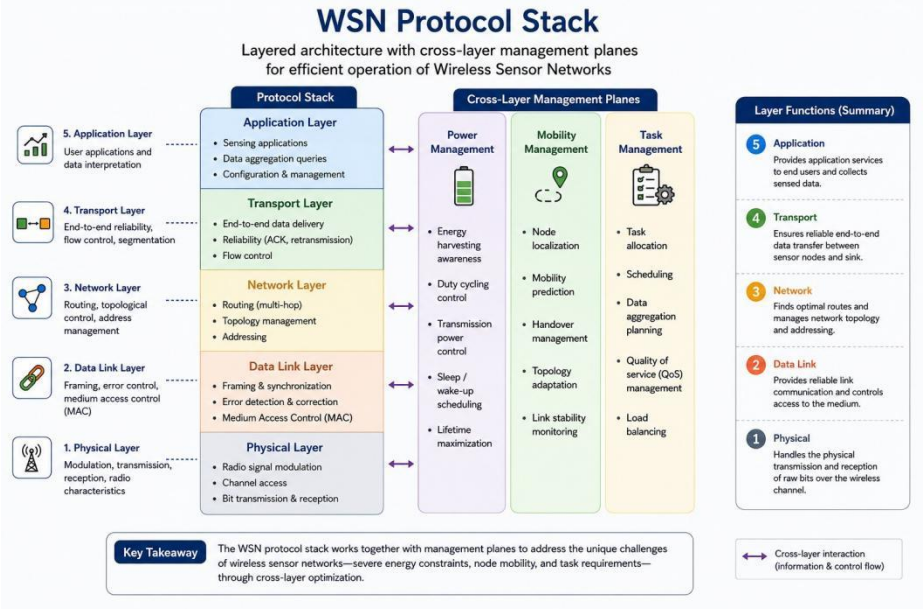

Figure 2.49: *WSN protocol stack — physical, data link, network, transport, and application layers, alongside power, mobility, and task management planes.*

2.6.1 Physical Layer

The **physical layer** is responsible for the actual transmission and reception of raw bits over the wireless medium. It handles tasks such as frequency selection, signal modulation, carrier frequency generation, and data encryption at the signal level. Common physical-layer technologies used in WSNs include **IEEE 802.15.4**, which underpins protocols such as **ZigBee**, as well as low-power variants of Bluetooth and proprietary sub-GHz radio standards.

2.6.2 Data Link Layer

The **data link layer** ensures reliable point-to-point and point-to-multipoint connections by managing **medium access control (MAC)**, error detection, and frame synchronisation. Since the wireless medium is shared among many nodes, the MAC sub-layer is critical in WSNs for avoiding collisions and minimising idle listening, both of which directly affect energy consumption. Common MAC strategies include **contention-based** protocols (such as CSMA/CA) and **schedule-based** protocols (such as TDMA), as well as hybrid approaches like **S-MAC** and **B-MAC**, which were specifically designed to reduce energy waste in sensor networks.

2.6.3 Network Layer

The **network layer** is responsible for **routing** — determining the path data packets take from source nodes to the base station. Given the energy constraints of WSNs, routing protocols at this layer are designed with energy efficiency as a primary objective, in contrast to traditional networks where throughput or latency often dominate design decisions. Notable WSN routing protocol families include **hierarchical/cluster-based** protocols (e.g., LEACH), **data-centric** protocols (e.g., Directed Diffusion, SPIN), and **location-based** protocols (e.g., GEAR), each suited to different application requirements.

2.6.4 Transport Layer

The **transport layer** provides mechanisms for reliable data delivery when required by the application, including segmentation, congestion control, and, in some protocols, limited retransmission. However, full end-to-end reliability schemes used in conventional networks (such as TCP) are often considered too resource-intensive for WSNs, so lightweight, application-aware transport protocols are typically employed instead.

2.6.5 Application Layer

The **application layer** sits at the top of the stack and is responsible for translating the network's raw sensing capabilities into meaningful services for the end user. This layer handles tasks such as data querying, network management, and interfacing with external software systems, and it defines how collected data is formatted, requested, and ultimately consumed by applications.

2.6.6 Cross-Layer Management Planes

In addition to the five primary layers, the WSN protocol stack incorporates three management planes that operate across all layers simultaneously:

- **Power management plane** — monitors and regulates the energy consumption of the node across all layers, for example by enabling sleep/wake scheduling and turning off the radio when not in use.
- **Mobility management plane** — detects and manages changes in node connectivity caused by node movement (in mobile WSN deployments) or node failure, helping maintain consistent routing paths.
- **Task management plane** — balances and schedules sensing and processing tasks among nodes within a region, helping to distribute energy consumption evenly and avoid overloading individual nodes.

2.7 Design Considerations of WSN Architecture

Designing an effective WSN architecture requires careful balancing of multiple, often competing, constraints. The unique nature of sensor nodes — small, inexpensive, and severely resource-limited — means that architectural decisions made at design time have a direct and lasting impact on network performance throughout its operational lifetime. The following factors are among the most critical design considerations.

- **Energy consumption and lifetime** — since most sensor nodes are battery-powered and often impossible to recharge after deployment, the architecture must minimise energy expenditure in sensing, processing, and especially communication, which is typically the most energy-costly operation.

- **Scalability** — the architecture must accommodate networks ranging from a handful of nodes to tens of thousands, without a significant degradation in performance, routing efficiency, or manageability as the network grows.
- **Fault tolerance and reliability** — given that individual nodes are prone to failure due to energy depletion, hardware faults, or environmental damage, the architecture must support graceful degradation and allow the network to reroute data and continue functioning despite the loss of individual nodes.
- **Network topology and deployment strategy** — the architecture must account for whether nodes are deployed deterministically (in fixed, planned positions) or randomly (scattered over an area), as this affects connectivity, coverage, and the routing strategies that can be employed.
- **Data aggregation and redundancy reduction** — since neighbouring nodes often sense highly correlated data, the architecture should support in-network aggregation to reduce redundant transmissions and conserve bandwidth and energy.
- **Quality of Service (QoS) requirements** — depending on the application, the architecture must satisfy specific requirements for latency, reliability, and data accuracy; for instance, healthcare or disaster-monitoring applications demand much stricter timeliness guarantees than long-term environmental studies.
- **Security** — the architecture must consider mechanisms for protecting data confidentiality, integrity, and authenticity, particularly in sensitive applications such as military surveillance or healthcare monitoring, while remaining mindful of the limited computational resources available for cryptographic operations.
- **Heterogeneity of nodes** — many practical deployments mix nodes with different capabilities (e.g., standard sensor nodes and more capable relay or gateway nodes); the architecture must support this heterogeneity efficiently rather than assuming uniform node capabilities.
- **Mobility support** — in applications where sensor nodes, the base station, or both may move (such as wildlife tracking or vehicular networks), the architecture must accommodate dynamic topology changes and maintain reliable connectivity.

2.8 Advantages of WSN Architecture

The architectural principles underlying Wireless Sensor Networks offer several significant advantages that have driven their widespread adoption across diverse application domains:

- **Flexible and rapid deployment** — since WSNs require no wired infrastructure, they can be deployed quickly across difficult or remote terrain, including areas that are otherwise inaccessible to conventional monitoring equipment.
- **Self-organisation and self-healing** — architectures designed for WSNs typically allow nodes to automatically discover neighbours, form routing structures, and reorganise the network when nodes fail or new nodes are added, all without manual intervention.
- **Energy-aware operation** — through architectural mechanisms such as clustering, data aggregation, and duty cycling, WSNs are able to significantly extend network lifetime despite the severe energy constraints of individual nodes.
- **Scalable coverage** — hierarchical and hybrid architectures, in particular, allow WSNs to scale from small local deployments to massive networks spanning large geographic regions, while keeping communication overhead manageable.
- **Cost efficiency** — the simplicity and low cost of individual sensor nodes, combined with the absence of wiring infrastructure, make WSN architectures considerably cheaper to deploy and maintain than traditional wired sensing systems.
- **Robustness through redundancy** — the dense deployment of nodes often results in overlapping sensing coverage, meaning that the failure of individual nodes does not necessarily result in loss of monitoring capability for a given area.
- **Adaptability to diverse applications** — the modularity of WSN architecture, particularly the separation between sensing, communication, and application layers, allows the same underlying architectural principles to be applied across vastly different domains, from agriculture to military surveillance.

2.9 Challenges of WSN Architecture

Despite their many advantages, WSN architectures must contend with a range of inherent challenges that stem largely from the resource-constrained nature of sensor nodes and the unpredictable environments in which they often operate.

- **Limited energy resources** — the most fundamental challenge in WSN architecture remains energy constraint; since replacing or recharging batteries in large-scale or inaccessible deployments is often impractical, every architectural decision must be evaluated against its energy cost.
- **Limited computational and memory resources** — sensor nodes typically possess modest processing power and very limited memory, restricting the complexity of algorithms, encryption schemes, and protocols that can be implemented on-node.
- **Unreliable wireless communication** — wireless links in WSNs are prone to interference, signal attenuation, multipath fading, and packet loss, particularly in harsh or obstructed environments, making reliable data delivery architecturally challenging.
- **Network scalability versus efficiency trade-off** — while hierarchical architectures improve scalability, they introduce additional complexity in cluster formation and management; as networks grow larger, maintaining both efficiency and simplicity becomes increasingly difficult.
- **Security vulnerabilities** — the open and shared nature of the wireless medium, combined with the limited capacity of nodes to perform robust cryptographic operations, exposes WSNs to a variety of attacks, including eavesdropping, node tampering, denial-of-service, and routing attacks such as sinkhole and wormhole attacks.
- **Node failure and fault management** — since individual node failure is common and expected in WSNs, the architecture must continuously detect, isolate, and route around failed nodes, which adds ongoing overhead to network operation.
- **Dynamic topology management** — in deployments involving node or base station mobility, or where nodes are added and removed over time, maintaining a consistent and efficient network topology becomes considerably more complex than in static networks.
- **Data redundancy and congestion** — densely deployed nodes often produce highly correlated or redundant data, which, if not properly managed through aggregation, can lead to

network congestion, increased latency, and unnecessary energy expenditure.

- **Standardisation and interoperability** — the diversity of hardware platforms, communication protocols, and proprietary systems across WSN vendors can hinder interoperability, complicating the integration of heterogeneous networks into unified systems such as IoT platforms.

Conclusion

This chapter has presented a comprehensive overview of Wireless Sensor Network architecture, beginning with a foundational definition of WSNs and an exploration of their growing importance across environmental, medical, agricultural, industrial, military, and urban application domains. The basic components that constitute a WSN — the sensor node, the base station, the communication network, and the task manager — were examined in detail, establishing the building blocks upon which larger architectural structures are built.

The chapter then explored the principal network architectures employed in WSNs, namely flat, hierarchical, and hybrid topologies, each offering distinct trade-offs between simplicity, scalability, and energy efficiency. The end-to-end data flow within a WSN was traced from initial sensing through local processing, routing, and final aggregation at the base station, followed by a detailed examination of the layered protocol stack that governs communication at every level of the network.

Finally, the chapter addressed the key design considerations that must guide architectural decisions, alongside the principal advantages that have made WSNs a transformative technology, and the persistent challenges — chiefly energy constraints, security vulnerabilities, and reliability issues — that continue to drive research and innovation in the field. Together, these elements establish the architectural foundation upon which the subsequent chapters of this project will build, particularly with respect to the specific protocols, algorithms, and design choices adopted in the proposed system.

Chapter 3

LabVIEW Program Integration

Introduction

LabVIEW (**Laboratory Virtual Instrument Engineering Workbench**) is a system-design and development environment created by **National Instruments (NI)**, now known as **NI**. It is built around a graphical programming language known as **G**, which allows engineers and scientists to create programs — referred to as **Virtual Instruments (VIs)** — by wiring together functional nodes on a block diagram rather than writing conventional text-based code. LabVIEW is widely used in academia, research institutions, and industry for applications involving data acquisition, instrument control, real-time monitoring, signal processing, and automated testing.

In the context of **Wireless Sensor Networks (WSNs)**, LabVIEW serves as a powerful supervisory platform that collects, processes, visualises, and logs sensor data received from distributed wireless nodes. By leveraging its built-in hardware drivers, communication libraries, and graphical user interface tools, LabVIEW enables the development of sophisticated monitoring dashboards and control systems without requiring expertise in low-level programming languages.

3.1 Overview of LabVIEW

3.1.1 History and Development

LabVIEW was first released in **1986** by National Instruments for the Apple Macintosh platform. Its introduction marked a paradigm shift in engineering software development, replacing command-line and text-based programming with an intuitive, dataflow-driven graphical approach. Over the decades, LabVIEW evolved to support a wide range of operating systems, including Windows, Linux, and macOS, and expanded its capabilities to include real-time systems, field-programmable gate arrays (FPGAs), embedded targets, and networked data acquisition systems.

Today, LabVIEW is one of the most widely adopted engineering development environments in the world, with applications spanning aerospace, automotive, biomedical engineering, telecommunications, and industrial automation.

3.1.2 LabVIEW Programming Model

LabVIEW programs are called **Virtual Instruments (VIs)** because they mimic the appearance and behaviour of physical instruments. Each VI consists of two primary components:

- **Front Panel** — the graphical user interface (GUI) of the VI, which contains controls (inputs) and indicators (outputs) such as knobs, buttons, charts, graphs, LEDs, and numeric displays. It represents what the end user interacts with during program execution.
- **Block Diagram** — the underlying program logic written using the G graphical language. It contains function nodes, constants, structures, and wires that define the data flow and computational operations of the VI.

LabVIEW follows a **dataflow programming model**, meaning that a node executes only when all of its required input data is available. This model naturally supports **parallel execution** of independent code paths, making LabVIEW particularly well-suited for real-time data acquisition and multi-channel monitoring applications.

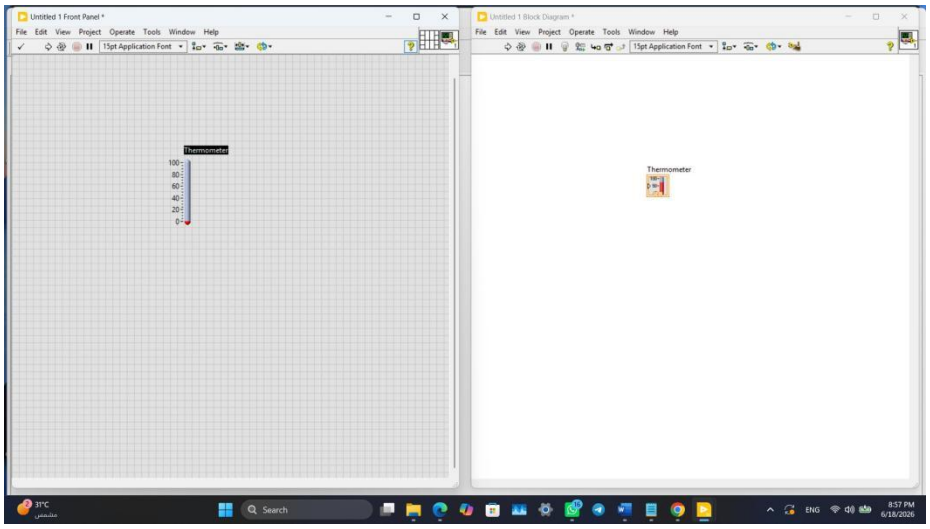

Figure 3.50: *LabVIEW programming environment showing the Front Panel (left) and Block Diagram (right) of a Virtual Instrument (VI).*

3.2 LabVIEW Key Components and Features

3.2.1 Virtual Instruments (VIs)

A **Virtual Instrument (VI)** is the fundamental building block of any LabVIEW program. VIs can be used as standalone applications or as **SubVIs** — reusable modular components embedded within other VIs, analogous to functions or subroutines in text-based programming languages. This modular architecture promotes code reusability, simplifies debugging, and improves the overall maintainability of large projects.

Key attributes of a VI include:

- A **Front Panel** for user interaction and data display.
- A **Block Diagram** containing the graphical source code.
- A **Connector Pane** that defines the inputs and outputs when the VI is used as a SubVI.
- An **Icon** that visually represents the VI on the block diagram of a parent VI.

3.2.2 Data Types and Wires

In LabVIEW, data is transmitted between nodes through **wires**. The colour and style of a wire indicate the data type it carries:

- **Orange** — floating-point numeric values (e.g., doubles).
- **Blue** — integer numeric values.
- **Green** — Boolean values (True/False).
- **Pink** — strings (text data).
- **Brown** — clusters (grouped data structures).
- **Dashed wires** — arrays of the corresponding data type.

3.2.3 Structures

LabVIEW provides graphical programming structures to control the flow and execution of code:

- **While Loop** — repeats execution of the enclosed code until a stop condition is met. Commonly used for continuous data acquisition and real-time monitoring.

- **For Loop** — executes the enclosed code a fixed number of times, determined by the iteration count input (N).
- **Case Structure** — implements conditional logic equivalent to an `if-else` or `switch` statement in text-based languages.
- **Sequence Structure** — enforces a specific sequential execution order of code sections, overriding the default dataflow behaviour.
- **Event Structure** — responds to user interface events (e.g., button clicks, value changes) in a non-polling, event-driven manner.

3.2.4 DAQ (Data Acquisition) Subsystem

The **DAQ subsystem** is one of LabVIEW's most powerful and widely used features. It provides a standardised software interface for communicating with physical data acquisition hardware, including NI DAQ boards, sensors, and signal conditioning modules. Key capabilities include:

- **Analog Input (AI)** — reading voltage or current signals from physical sensors such as temperature probes, pressure sensors, and accelerometers.
- **Analog Output (AO)** — generating analog control signals to drive actuators, motors, or other external equipment.
- **Digital Input/Output (DIO)** — reading and writing digital logic signals for switching, triggering, and control applications.
- **Counter/Timer** — measuring frequency, counting pulses, and generating timed signals.

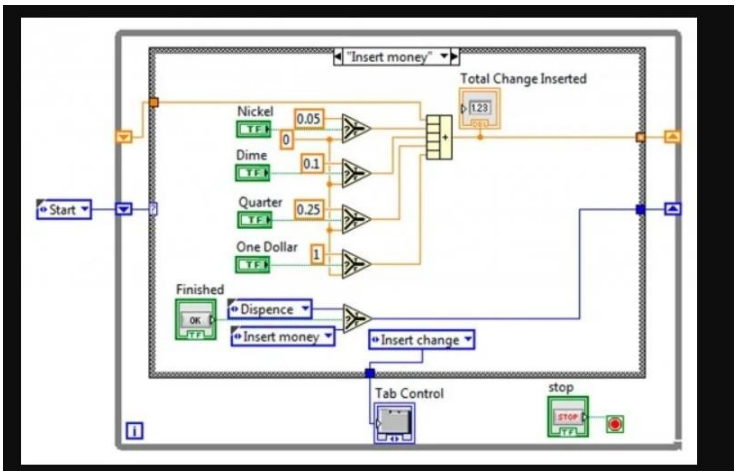

Figure 3.51: *LabVIEW DAQ Assistant — a graphical configuration tool for setting up data acquisition tasks from physical hardware.*

3.3 LabVIEW in Wireless Sensor Networks

3.3.1 Role of LabVIEW in WSN Applications

In a Wireless Sensor Network, distributed sensor nodes continuously collect physical measurements — such as temperature, humidity, pressure, light intensity, or gas concentration — and transmit the data wirelessly to a central station. LabVIEW acts as the **supervisory software layer** at the central station, responsible for:

- **Data Reception** — receiving data packets transmitted wirelessly from sensor nodes via communication interfaces such as serial (UART), USB, Ethernet, or dedicated wireless modules (e.g., XBee, Zigbee, Wi-Fi).
- **Data Parsing and Decoding** — extracting raw sensor values from incoming data frames, applying calibration offsets, and converting raw ADC readings into physical engineering units.
- **Real-Time Visualisation** — displaying live sensor readings on graphical indicators, waveform charts, and gauges on the LabVIEW Front Panel.
- **Data Logging** — recording sensor measurements to local files (CSV, TDMS) or databases for historical analysis and reporting.
- **Alarm and Threshold Management** — triggering alerts or automated responses when sensor values exceed predefined safety thresholds.
- **Control Output** — sending commands back to remote nodes or actuators based on the processed sensor data (closed-loop control).

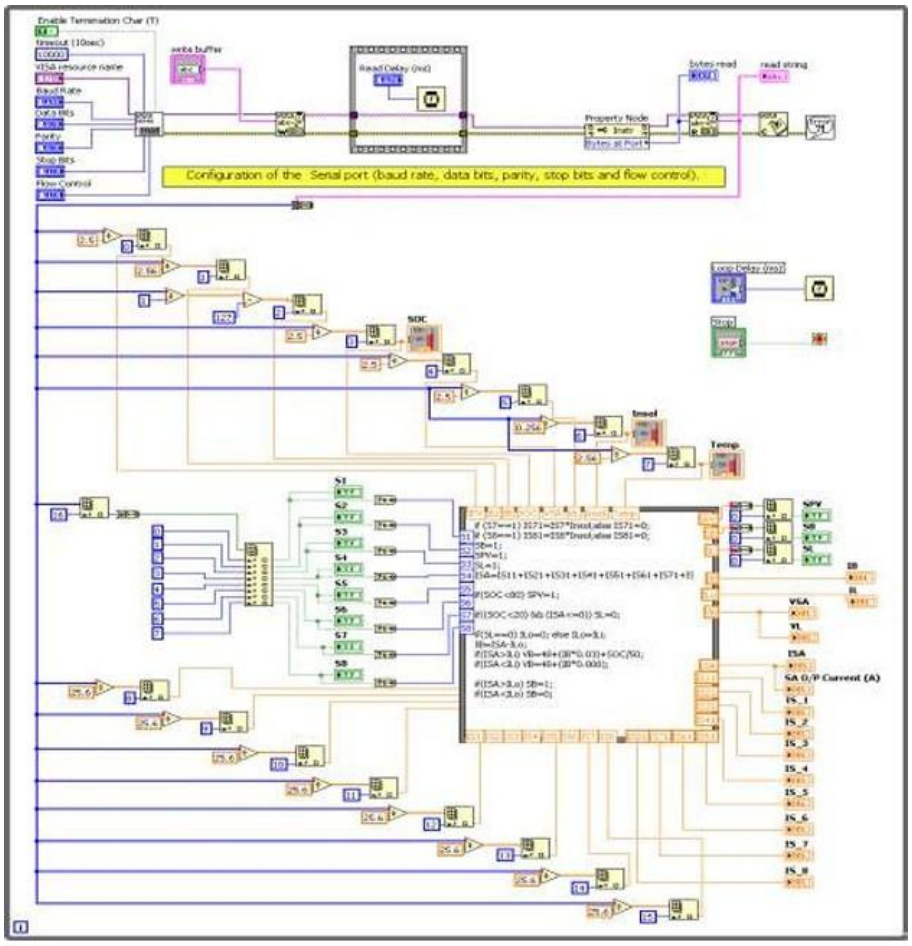

Figure 7, Monitoring System for PV Power Station Block Diagram.

Figure 3.52: *LabVIEW as the central monitoring platform in a Wireless Sensor Network architecture, interfacing with distributed sensor nodes.*

3.3.2 Communication Interfaces

LabVIEW supports a wide variety of communication protocols and hardware interfaces that are essential for integrating with WSN nodes:

Serial Communication (UART / RS-232 / RS-485)

Serial communication is one of the most common methods for interfacing sensor nodes with LabVIEW. Microcontroller-based nodes transmit data frames over a UART interface to a host computer via a USB-to-serial adapter. LabVIEW's **VISA (Virtual Instrument Software Architecture)** library provides comprehensive functions for configuring baud rate, data bits, parity, and stop bits, as well as reading and writing serial data.

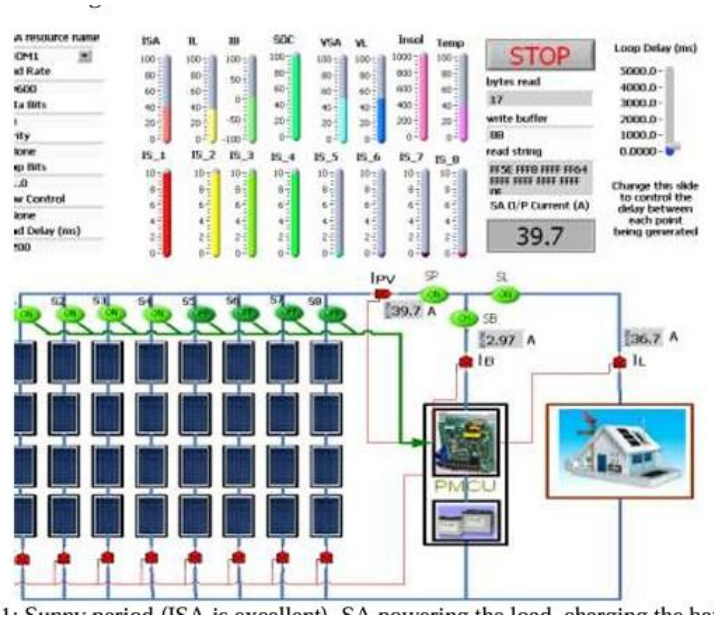

Figure 3.53: *LabVIEW VISA Serial communication block diagram for reading data from a sensor node via UART.*

TCP/IP and UDP Communication

For WSN systems where sensor nodes are connected over an IP-based network (Ethernet or Wi-Fi), LabVIEW provides native **TCP** and **UDP** communication functions. These allow LabVIEW to act as a server or client, establishing socket connections with remote nodes to exchange data packets in real time.

- **TCP (Transmission Control Protocol)** — used when reliable, ordered data delivery is required, such as for configuration commands or alarm messages.
- **UDP (User Datagram Protocol)** — used when low-latency, high-frequency data streaming is preferred over guaranteed delivery, such as for sensor telemetry.

Wireless Module Integration (XBee / Zigbee / Wi-Fi)

In many WSN deployments, sensor nodes communicate wirelessly using dedicated radio modules. LabVIEW interfaces with these modules via serial or USB connections at the base station side. Protocols such as **Zigbee** (IEEE 802.15.4) are commonly used due to their low power consumption and mesh networking capability. LabVIEW reads the incoming wireless data frames and parses them into structured sensor readings.

3.4 Data Acquisition and Processing in LabVIEW

3.4.1 Data Acquisition Pipeline

The data acquisition pipeline in a LabVIEW-based WSN monitoring system typically follows a structured flow of operations, executed continuously within a **While Loop**:

1. **Initialisation** — configure communication ports, define sampling parameters, and initialise data storage structures.
2. **Data Reception** — read incoming data frames from the wireless sensor nodes via the selected communication interface.
3. **Parsing and Validation** — extract individual sensor values from the received frame; validate data integrity using checksum or CRC verification.
4. **Unit Conversion and Calibration** — apply calibration coefficients and scaling factors to convert raw digital values into physical engineering units (e.g., °C, %, kPa).
5. **Visualisation** — update waveform charts, numeric indicators, and graphical displays on the Front Panel.
6. **Threshold Checking** — compare measured values against predefined limits and trigger alarms or control outputs if thresholds are exceeded.
7. **Data Logging** — write timestamped readings to a file or database for offline analysis.
8. **Loop Iteration** — repeat from step 2 at the specified sampling interval until the stop condition is activated.

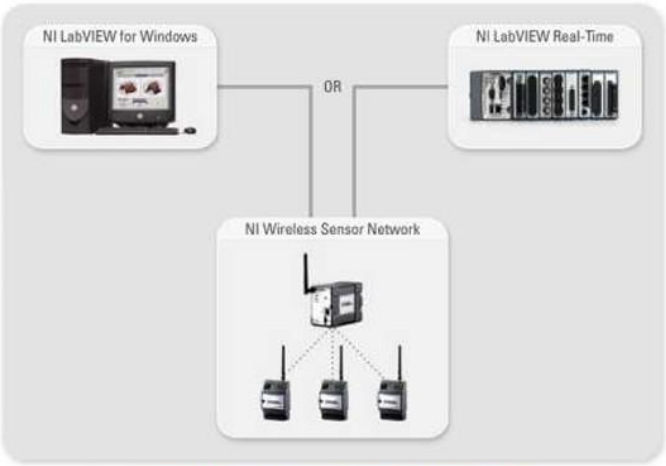
NI LabVIEW for Windows
OR
NI LabVIEW Real-Time
NI Wireless Sensor Network
Figure 14, Real time and windows based options [14]

Figure 3.54: *LabVIEW Block Diagram illustrating the complete data acquisition pipeline for WSN monitoring.*

3.4.2 Signal Processing

LabVIEW includes a comprehensive suite of built-in signal processing functions, accessible through its **Signal Processing** palette. These functions allow raw sensor data to be conditioned and analysed in real time:

- **Filtering** — applies digital filters (low-pass, high-pass, band-pass) to remove noise and interference from sensor signals.
- **Statistical Analysis** — computes mean, standard deviation, minimum, maximum, and root mean square (RMS) values over a sliding window of samples.
- **Fast Fourier Transform (FFT)** — transforms time-domain sensor data into the frequency domain for spectral analysis of vibration, acoustic, or electrical signals.
- **Waveform Alignment** — synchronises data streams from multiple sensor channels for accurate comparative analysis.
- **Peak Detection** — identifies significant peaks or anomalies in sensor data that may indicate fault conditions.

3.4.3 Data Logging and File I/O

LabVIEW supports multiple file formats for storing acquired sensor data:

- **TDMS (Technical Data Management Streaming)** — NI's proprietary binary format optimised for high-speed data logging, supporting metadata, channel descriptions, and timestamps. Readable by LabVIEW, DIAdem, and Microsoft Excel via a free plugin.
- **CSV (Comma-Separated Values)** — a plain-text format compatible with Microsoft Excel, MATLAB, and Python for post-processing and reporting.
- **Text Files** — used for logging event records, alarm histories, and system status messages.
- **Database Connectivity** — LabVIEW can log data directly to SQL databases (e.g., MySQL, Microsoft SQL Server) via ODBC or the Database Connectivity Toolkit.

Figure 3.55: *LabVIEW data logging VI writing timestamped sensor readings to a TDMS file.*

3.5 LabVIEW Front Panel Design for WSN Monitoring

3.5.1 User Interface Components

The LabVIEW Front Panel serves as the **Human-Machine Interface (HMI)** of the monitoring system, presenting real-time sensor data in a clear and actionable format. The following graphical controls and indicators are typically used in a WSN monitoring dashboard:

- **Waveform Chart** — displays a continuously scrolling time-series plot of sensor values as new data arrives, providing a live view of signal trends over time.
- **Waveform Graph** — plots a complete array of data points at once, suitable for displaying a fixed-length snapshot of sensor measurements.
- **Numeric Indicator** — shows the current value of a sensor channel as a digital numerical readout.
- **Thermometer / Gauge / Meter** — provides an analogue-style visual representation of a measured quantity, making it easy to assess the value relative to its operating range at a glance.
- **LED Indicator** — a Boolean indicator that illuminates to signal alarm states, connectivity

status, or threshold violations.

- **String Indicator** — displays text-based status messages, error descriptions, or received data frames.
- **Boolean Button (Stop / Start)** — interactive control elements allowing the operator to start or stop data acquisition, trigger calibration routines, or reset alarm states.
- **Tab Control** — organises multiple panels within a single Front Panel window, grouping related sensor channels or functional sections for a cleaner interface layout.

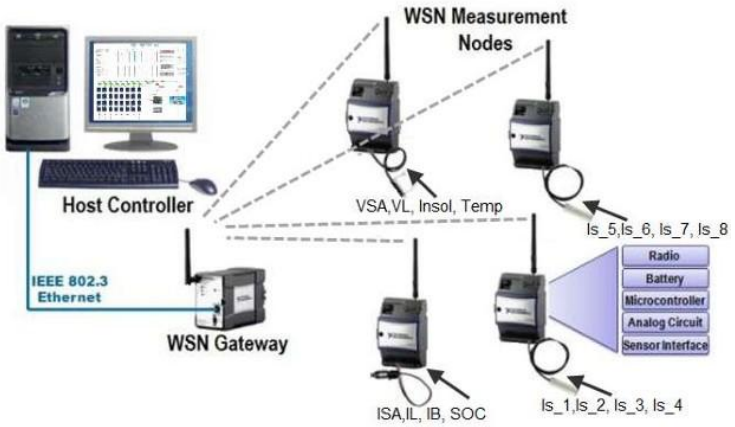

Figure 15, Typical integration of WSN 3202 and NI WSN-9791 modules with monitoring system [15].

Figure 3.56: *LabVIEW Front Panel dashboard for real-time WSN monitoring, displaying waveform charts, numeric indicators, and alarm LEDs.*

3.5.2 Alarm and Threshold Management

A critical feature of any sensor monitoring system is the ability to automatically detect and respond to abnormal conditions. In LabVIEW, alarm management is implemented by comparing live sensor readings against predefined upper and lower threshold limits using **Comparison** functions on the block diagram. When a threshold is violated, the system can:

- Illuminate an LED alarm indicator on the Front Panel.
- Display a pop-up warning message using a **One Button Dialog** or **User Event**.
- Log the alarm event with a timestamp to a dedicated alarm log file.
- Activate a digital output signal to trigger an external buzzer, relay, or shutdown mechanism.
- Transmit a command packet back to the sensor node to adjust its operating parameters.

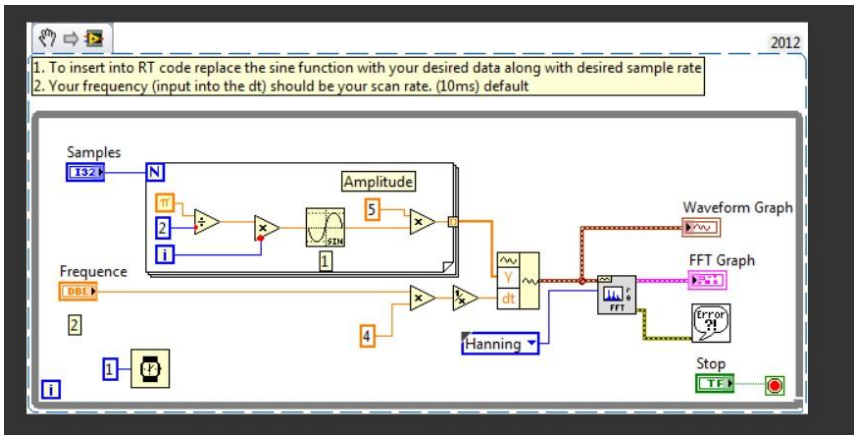
2012
1. To insert into RT code replace the sine function with your desired data along with desired sample rate
2. Your frequency (input into the dt) should be your scan rate. (10ms) default
Samples
Amplitude
Frequence
Waveform Graph
FFT Graph
Hanning
Stop

Figure 3.57: *LabVIEW alarm management block diagram, comparing sensor readings against threshold limits and triggering alarm indicators.*

3.6 Integration of LabVIEW with the WSN System

3.6.1 System Architecture

In this project, LabVIEW is integrated as the **central monitoring and control application** within the WSN system. The overall architecture consists of three main tiers:

1. **Sensing Tier** — distributed wireless sensor nodes equipped with environmental sensors (e.g., temperature, humidity) and a microcontroller that samples the sensors and transmits data wirelessly.
2. **Communication Tier** — the wireless network infrastructure responsible for relaying sensor data from the nodes to the base station, using a protocol suited to the deployment environment.
3. **Application Tier** — the LabVIEW application running on the host computer at the base station, which receives, processes, displays, and logs all incoming sensor data.

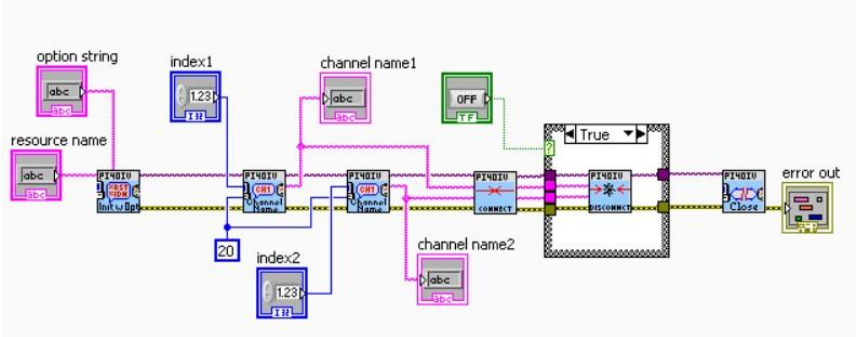

Figure 3.58: *Three-tier WSN system architecture with LabVIEW at the application layer.*

3.6.2 Data Flow in the Integrated System

The complete data flow from a sensor node to the LabVIEW monitoring application proceeds as follows:

1. A sensor node measures a physical quantity (e.g., ambient temperature) using the appropriate transducer.
2. The onboard microcontroller samples the sensor output via its ADC, applies local calibration if required, and formats the measurement into a structured data frame (e.g., ASCII string or binary packet).
3. The data frame is transmitted wirelessly to the base station receiver using the designated wireless protocol.
4. The base station receiver forwards the data to the host computer via a serial (USB/UART) or Ethernet interface.
5. The LabVIEW VI reads the incoming data, parses the frame, extracts the sensor values, and processes them through the acquisition pipeline.
6. The processed data is displayed on the Front Panel dashboard and written to the data log file with a timestamp.

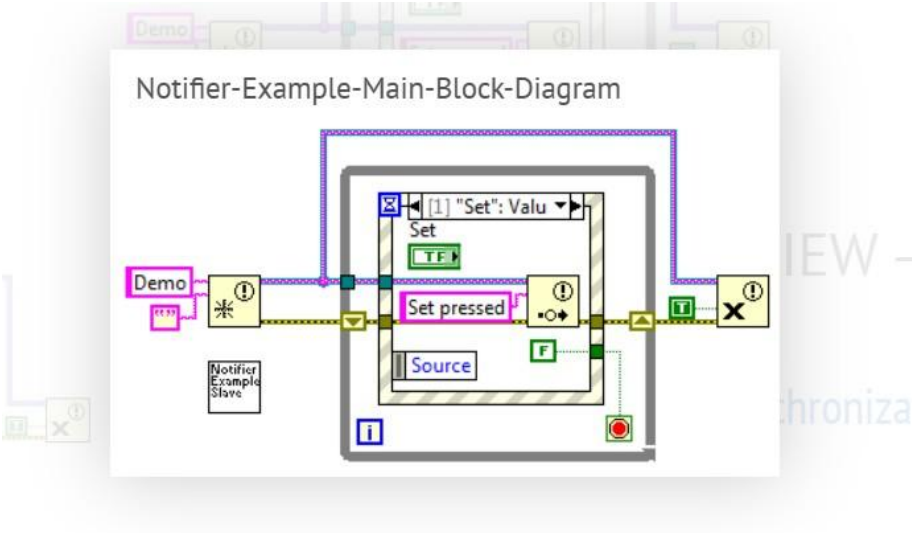

Figure 3.59: *End-to-end data flow from WSN sensor node to LabVIEW monitoring application.*

3.6.3 LabVIEW VI Structure for This Project

The LabVIEW application developed in this project is structured as a hierarchy of VIs, each responsible for a specific functional aspect of the system:

- **Main VI** — the top-level VI containing the main While Loop, the Front Panel dashboard, and the orchestration logic that calls all SubVIs in the correct sequence.
- **Communication SubVI** — handles the opening, configuration, reading, and closing of the serial or network communication port used to receive data from the wireless sensor nodes.
- **Data Parser SubVI** — receives a raw data string or byte array and extracts individual sensor channel values, validates the frame integrity, and returns structured numeric outputs.
- **Unit Conversion SubVI** — applies calibration coefficients and scaling equations to convert raw ADC values into physical engineering units.
- **Alarm Handler SubVI** — compares current sensor readings against configurable thresholds and returns Boolean alarm flags for each channel.
- **Data Logger SubVI** — appends a new row of timestamped sensor readings to the designated log file in CSV or TDMS format.

3.7 Advantages of Using LabVIEW in WSN Projects

The selection of LabVIEW as the monitoring and data acquisition platform for this project is justified by several key advantages that it offers over alternative approaches:

- **Graphical Development Environment** — LabVIEW's G programming language eliminates the need for complex text-based code, allowing engineers to design, modify, and debug data acquisition systems rapidly and intuitively.
- **Native Hardware Support** — LabVIEW provides extensive built-in drivers for a wide range of data acquisition boards, sensors, and communication interfaces, significantly reducing integration time.
- **Real-Time Execution** — LabVIEW's dataflow model and multithreaded execution engine are inherently suited to parallel real-time tasks, enabling simultaneous data acquisition, processing, and display without additional concurrency management.
- **Rich Signal Processing Library** — the built-in signal analysis and processing palettes allow advanced operations such as filtering, FFT, and statistical analysis to be applied directly within the monitoring application.
- **Flexible Data Logging** — LabVIEW supports multiple file formats for data logging (TDMS, CSV, databases), making it easy to export data for post-processing in external tools such as MATLAB, Python, or Excel.
- **Scalability** — a LabVIEW-based WSN monitoring system can be readily scaled to accommodate additional sensor nodes, channels, or communication protocols by extending the VI hierarchy with new SubVIs.
- **Cross-Platform Deployment** — LabVIEW applications can be compiled as standalone executables for deployment on Windows, Linux, and real-time embedded targets without requiring a full LabVIEW installation on the host machine.
- **Industry Standard** — LabVIEW is widely recognised and adopted in academic research, industrial automation, and scientific instrumentation, making it a professional and well-supported choice for this type of project.

Chapter 4

Project Problem Formulation

Introduction

The design of a Wireless Sensor Network (WSN) for real-time monitoring of a Solar Photovoltaic/Thermal (PV/T) system requires a rigorous mathematical formulation of the network's structure, objectives, and operational constraints. This section presents the formal problem formulation governing the deployment of the **NI WSN-9791 Gateway** and the **NI WSN-3212 / NI WSN-3202** sensor nodes, establishing the theoretical foundation upon which the routing, scheduling, and energy-management decisions of the system are based.

4.1 Motivation for a Formal Problem Statement

Monitoring a PV/T collector is fundamentally different from monitoring a generic industrial process, because the physical quantities of interest — absorber-plate temperature, ambient temperature, and the voltage/current output of the photovoltaic cells — are continuously time-varying, weather dependent, and safety relevant. An under-sampled or delayed reading may allow a thermal hotspot or an over-voltage condition to go undetected, while an over-sampled or poorly scheduled network wastes the limited battery energy of the NI WSN-3212/3202 nodes and increases the probability of radio-channel contention. A purely qualitative design — i.e., simply placing nodes on the collector and configuring default polling rates — cannot answer the quantitative questions that determine whether the deployed system is fit for purpose:

- What is the expected operational lifetime of the network before the first node requires battery replacement?
- How many nodes can the NI WSN-9791 gateway reliably service without violating the channel-capacity or collision constraints?
- What sampling interval guarantees that a thermal transient on the PV/T absorber is not missed, while still respecting the energy budget of the NI WSN-3212 node?
- What is the worst-case latency between a sensor reading exceeding a safety threshold and the corresponding alert being delivered through the Python-based Telegram bot?

Answering these questions requires translating the physical deployment into a formal mathematical model, defining explicit objective functions, and enumerating the constraints imposed by the hardware. The remainder of this section develops that formulation in a layered man-

ner: the system model first (Section 3.2), followed by the objective functions (Section 3.3), the design constraints (Section 3.4), the decision variables (Section 3.5), a worked numerical instantiation of the model (Section 3.6), a sensitivity and trade-off discussion (Section 3.7), and finally a summary table that consolidates the complete notation used throughout the chapter (Section 3.8).

4.2 Hardware Reference Specifications

The mathematical formulation developed in this chapter is grounded in the physical characteristics of the NI WSN hardware platform actually used in the implementation. Table 4.5 summarises the parameters that recur throughout the subsequent derivations.

Table 4.5: *Reference hardware parameters used in the problem formulation.*

Component	Role in the WSN	Relevant Parameter
NI WSN-9791	Central Gateway; aggregates data from all nodes and interfaces with LabVIEW host	Channel capacity C_{ch}, maximum node count k_{max}
NI WSN-3212	4-channel thermocouple input node	Sampling interval Δt_T, battery capacity E_{max}
NI WSN-3202	Analog voltage/current input node	Sampling interval Δt_E, battery capacity E_{max}
Radio Interface	IEEE 802.15.4-compliant 2.4 GHz transceiver	Transmission range $R_{\max}$, receiver sensitivity $RSSI_{min}$

These parameters are treated as fixed, hardware-imposed boundaries within which the optimisation variables of Section 3.5 are free to operate.

4.3 System Model and Deployment

4.3.1 Network Topology Representation

The WSN is modelled as an undirected, connected graph

$$G = (V, E)$$

where $V = \{n_1, n_2, \ldots, n_k, GW\}$ denotes the set of deployed sensor nodes together with the central gateway node GW (the NI WSN-9791), and $E \subseteq V \times V$ represents the set of feasible wireless communication links between node pairs.

Each sensor node $n_i \in V$ is characterised by the tuple:

$$n_i = (p_i, E(t), s(t), \tau_i)$$

where:

- $p_i = (x_i, y_i, z_i)$ — the spatial coordinates of node n_i on the PV/T panel array, where the z_i component accounts for mounting elevation differences across the collector surface.
- $E(t)$ — the residual battery energy of node n_i at time t.
- $s(t) \in \{0, 1\}$ — the binary operational state of the node (1 = active, 0 = sleep).
- τ_i — the node type, $\tau_i \in \{3212(Thermocouple), 3202(Voltage/Current)\}$.

4.3.2 Spatial Deployment Model

The PV/T monitoring array is partitioned into k thermal and electrical monitoring zones. Each NI WSN-3212 node is assigned to a thermocouple cluster $C_T \subset V$ responsible for surface-temperature acquisition across the thermal absorber plate, while each NI WSN-3202 node is assigned to an electrical cluster $C_E \subset V$ responsible for PV cell voltage and current sampling, such that:

$$V = C_T \cup C_E \cup \{GW\}, \quad C_T \cap C_E =$$

based NI WSN radio:

$$(n_i, n_j) \in E \iff p_i - p_j \leq R_{\max}$$

Given the star/tree topology adopted by the NI WSN-9791 gateway, every node n_i communicates with the gateway either directly (single-hop) or through an intermediate relay node n_r (multi-hop), defining a routing path:

$$Path(n_i \rightarrow GW) = \{n_i, n_{i+1}, \ldots, n_r, GW\}$$

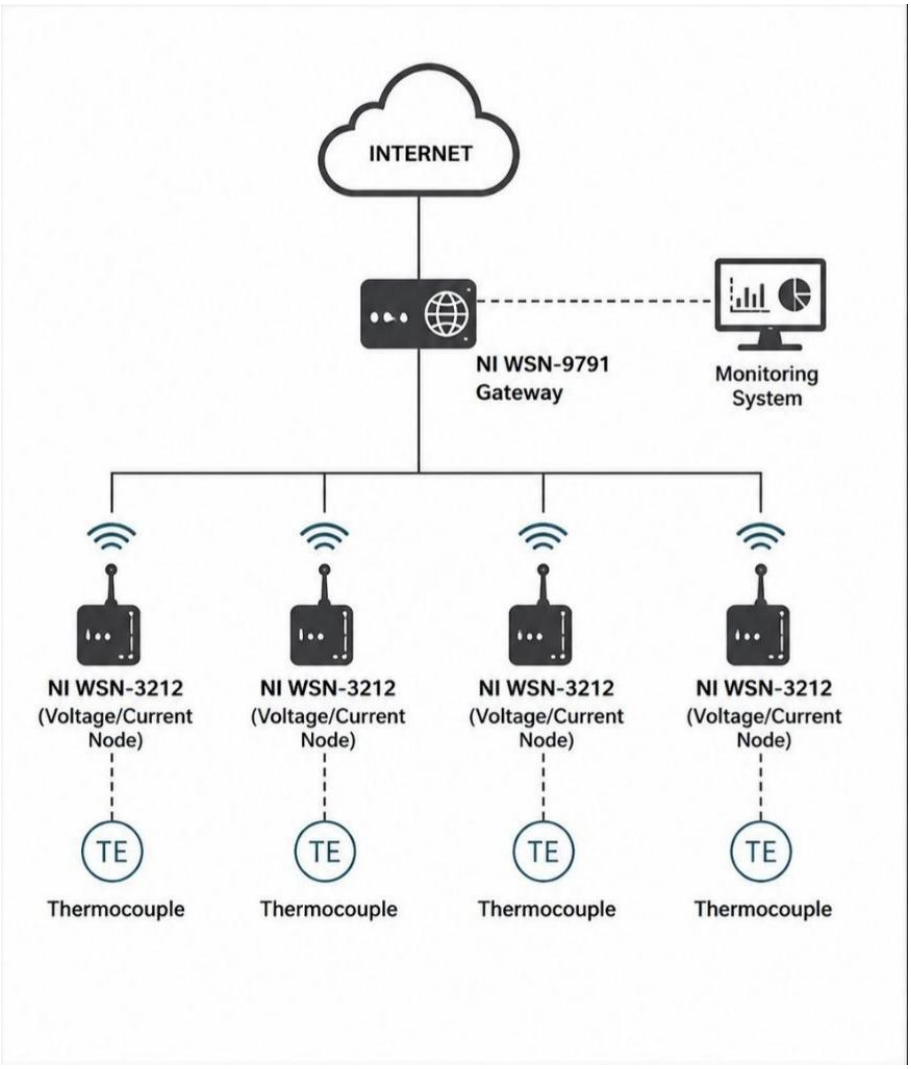

Figure 4.62: *Logical deployment model of the NI WSN-9791 Gateway with NI WSN-3212 (thermocouple) and NI WSN-3202 (voltage/current) nodes across the PV/T monitoring array.*

4.3.3 Adjacency and Connectivity Representation

For analytical and simulation purposes, the topology graph $G = (V, E)$ is equivalently represented by a $(k + 1) \times (k + 1)$ **adjacency matrix** A, where $k = |C_T| + |C_E|$ is the total number of deployed sensor nodes (excluding the gateway):

$$A_{ij} = \{1 \text{ , } if(n_i, n_j) \in EO, otherwise$$

The matrix A is symmetric ($A_{ij} = A_{ji}$) because the wireless links established by the NI WSN radios are bidirectional. The **network connectivity** requirement — i.e., the condition that every deployed node has at least one feasible path to the gateway — is formally expressed using the graph Laplacian $L = D - A$, where D is the diagonal degree matrix. The network is connected if and only if the second-smallest eigenvalue of L (the *algebraic connectivity*, λ_2) satisfies:

$$\lambda_2(L) > 0$$

This condition is used during the deployment-planning phase to verify, prior to physical installation, that the proposed placement of NI WSN-3212/3202 nodes around the PV/T collector does not leave any node isolated from the NI WSN-9791 gateway.

4.3.4 Node Degree and Gateway Load

The **degree** of the gateway node, $deg(GW)$, is the number of nodes that communicate with it directly (single-hop):

$$deg(GW) = \sum_{i=1}^{k} A_{i,GW}$$

Because the NI WSN-9791 gateway services every node in the network — either directly or as the terminus of a multi-hop path — its effective load is bounded by the total number of deployed nodes:

$$deg(GW) \leq k \leq k_{max}$$

where k_{max} is the maximum number of nodes the gateway can address, as constrained by its addressing scheme and channel capacity (see Section 3.4.3). This bound is particularly relevant for the PV/T application, since the thermocouple cluster $\mathcal{C}_T$ typically requires a higher node density than the electrical cluster $\mathcal{C}_E$ in order to resolve spatial temperature gradients across the absorber plate.

4.3.5 Rationale for Cluster-Based Spatial Partitioning

The partitioning of V into $\mathcal{C}_T$ and $\mathcal{C}_E$, introduced in Section 3.2.2, is not an arbitrary labelling but a direct consequence of the differing spatial and temporal characteristics of the two physical quantities being monitored:

- **Thermal cluster $\mathcal{C}_T$.** Surface temperature on a PV/T absorber plate varies continuously and smoothly in space; however, localised effects such as partial shading, soldering defects, or coolant-flow non-uniformity can produce sharp local gradients. The NI WSN-3212 thermocouple nodes are therefore distributed so that no two adjacent thermocouple positions are separated by more than a maximum thermal-gradient resolution distance d_{res}^{T}.

- **Electrical cluster $\mathcal{C}_E$.** Voltage and current are effectively uniform across a series-connected string of PV cells under normal operation, so the NI WSN-3202 nodes are deployed sparsely, typically one node per string or per sub-array, rather than at the same density as the thermal cluster.

This distinction directly affects the optimisation problem: $|\mathcal{C}_T| \gg |\mathcal{C}_E|$ in most practical PV/T installations, meaning that the thermal cluster dominates both the energy budget (Section 3.3.2) and the channel-occupancy constraint (Section 3.4.3) of the network.

4.4 Objective Functions

The WSN design problem is fundamentally a multi-objective optimisation problem, balancing energy efficiency, network longevity, and data reliability. The three primary objective functions are formulated below.

4.4.1 Objective 1 — Maximisation of Network Lifetime

Network lifetime T_{life} is defined as the time elapsed until the first critical node (commonly referred to as the **First Node Death, FND**) depletes its battery energy below the minimum operational threshold E_{th}:

$$T_{life} = \min_{i \in V} \left\{ t \mid E_i(t) \leq E_{th} \right\}$$

The optimisation goal is therefore expressed as:

$$\max \; T_{life} = \max_{i \in V} \left(\min \left\{ t \mid E_i(t) \leq E_{th} \right\} \right)$$

This formulation ensures that the monitoring system continues to provide continuous, uninterrupted thermal and electrical readings of the PV/T array for the longest possible duration before requiring battery replacement or recharging.

Alternative Lifetime Metrics

While the FND criterion is the most conservative and is adopted as the primary lifetime metric in this work, two complementary metrics are also defined to characterise the degradation behaviour of the network over time, and are reported alongside T_{life} in the simulation results chapter:

- **Half Node Death (HND)** — the time at which 50% of the deployed nodes have depleted their energy below E_{th}:

$$T_{HND} = \min \left\{ t \mid |\{n_i \in V : E_i(t) \leq E_{th}\}| \geq k2 \right\}$$

- **Last Node Death (LND)** — the time at which the final surviving node is depleted, repre-

senting the absolute upper bound on the network's operational coverage:

$$T_{LND} = \max_{i \in V} \{ t \mid E_i(t) \leq E_{th} \}$$

For a PV/T monitoring application, the FND metric is the most operationally meaningful, because the loss of even a single thermocouple node creates a spatial blind spot on the absorber plate that could conceal a localised hotspot; the system is therefore considered to have entered a *degraded monitoring state* as soon as FND occurs, even though the remaining nodes continue to operate.

Coverage-Aware Lifetime

Because the thermal cluster $\mathcal{C}_T$ provides spatial coverage of the PV/T surface, a stricter, coverage-aware definition of network lifetime is also introduced. Let $\Phi(t) \subseteq \mathcal{C}_T$ denote the set of thermocouple nodes still active at time t, and let κ_{min} denote the minimum number of active thermocouples required to resolve the thermal-gradient resolution requirement of Section 3.2.4. The **coverage-preserving lifetime** is:

$$T_{cov} = \min \{ t \mid |\Phi(t)| < \kappa_{min} \}$$

Since $T_{cov} \leq T_{LND}$ by construction, design decisions in this project prioritise T_{cov} over the raw LND metric whenever the two are in conflict.

4.4.2 Objective 2 — Minimisation of Energy Consumption

The total energy consumed by node n_i over a sensing-transmission cycle is the sum of the sensing, processing, and communication energy components:

$$E_i^{total} = E_i^{sense} + E_i^{proc} + E_i^{tx} + E_i^{rx}$$

Using the widely adopted **first-order radio energy dissipation model**, the transmission energy for an l-bit packet over distance d is given by:

$$E_{tx}(l, d) = \begin{cases} l \cdot E_{elec} + l \cdot \varepsilon_{fs} \cdot d^2, & d < d_0 \\ l \cdot E_{elec} + l \cdot \varepsilon_{mp} \cdot d^4, & d \geq d_0 \end{cases}$$

$$E_{rx}(l) = l \cdot E_{elec}$$

where:

- E_{elec} — energy dissipated per bit by the transmitter/receiver electronics.
- ε_{fs}, ε_{mp} — the free-space and multi-path fading amplification coefficients, respectively.
- $d_0 = \sqrt{\varepsilon_{fs}/\varepsilon_{mp}}$ — the crossover distance threshold.

The network-wide optimisation objective is to minimise the cumulative energy expenditure across all k nodes over the monitoring horizon T:

$$\min E_{network} = \sum_{i=1}^{k} \sum_{t=1}^{T} E_i^{total}(t$$

$$\min$$

Sensing and Processing Energy Components

In addition to the communication energy modelled above, each NI WSN-3212 thermocouple node consumes a measurable quantity of energy during the analog-to-digital conversion of the thermocouple voltage and during the cold-junction-compensation computation performed on-board. This is modelled as a fixed per-sample cost:

$$E_i^{sense} = e_{ADC} + e_{CJC}$$

where e_{ADC} is the energy required for one analog-to-digital conversion cycle and e_{CJC} is the energy required for the cold-junction-compensation correction applied to raw thermocouple readings. For the NI WSN-3202 node, $e_{CJC} = 0$ since no thermocouple linearisation is required, and the sensing energy reduces to the ADC conversion cost alone.

The processing energy E_i^{proc} accounts for the on-board microcontroller cycles required to packetise the sample, append the node identifier and timestamp, and apply any local threshold comparison used for alert generation (Section 3.4.3):

$$E_i^{proc} = e_{cyc} \cdot N_{cyc}$$

where e_{cyc} is the energy per clock cycle and N_{cyc} is the number of cycles required for the packetisation and threshold-check routine.

Illustrative Numerical Instantiation

To demonstrate the practical use of the energy model, Table 4.6 presents a representative numerical instantiation using typical first-order radio model parameters reported in the WSN literature ($E_{elec} = 50$ nJ/bit, $\varepsilon_{fs} = 10$ pJ/bit/m², $\varepsilon_{mp} = 0.0013$ pJ/bit/m⁴), together with an assumed packet size of $l = 64$ bytes $= 512$ bits, in order to illustrate the relative magnitude of the trans- mission energy at three representative distances between a sensor node and the NI WSN-9791 gateway.

Table 4.6: *Illustrative transmission energy per packet at varying node-to-gateway distance ($l = 512$ bits).*

Distance d (m)	Regime	E_{tx} (J)	Relative Cost
2	Free-space ($d < d_0$)	$\approx 4.61 \times 10^{-5}$	1× (baseline)
10	Free-space ($d < d_0$)	$\approx 7.68 \times 10^{-5}$	≈ 1.7×
30	Multi-path ($d \geq d_0$)	$\approx 6.13 \times 10^{-4}$	≈ 13.3×

The non-linear growth of E_{tx} beyond the crossover distance d_0 quantitatively justifies the multi- hop routing strategy introduced in Section 3.2.2: relaying data through an intermediate node at $d < d_0$ on each hop is, for sufficiently large total distances, more energy-efficient than a single long-range transmission directly to the gateway, despite the additional reception and re- transmission overhead at the relay.

4.4.3 Objective 3 — Maximisation of Data Reliability

Given the safety-critical nature of real-time alerting (e.g., over-temperature or over-current conditions triggering the Telegram bot notification), the **Packet Delivery Ratio (PDR)** must be maximised:

$$PDR = \frac{N_{received}}{N_{sent}} \times 100\%$$

The composite reliability objective additionally penalises end-to-end latency L_i for alert-class packets, since delayed delivery undermines the purpose of real-time monitoring:

$$\max R = \max \; w_1 \cdot PDR - w_2 \cdot \bar{L}, \; w_1 + w_2 = 1$$

where $\bar{L}$ is the average end-to-end latency and w_1, w_2 are weighting coefficients reflecting the relative importance of delivery success versus responsiveness for the application.

Link-Level Reliability Model

The end-to-end PDR of a multi-hop path is itself a function of the per-hop link reliability. If p_{ij} denotes the probability of successful packet delivery across a single link $(n_i, n_j) \in E$, then the probability that a packet originating at node n_i successfully reaches the gateway along path $Path(n_i \rightarrow GW)$ of length $H(n_i)$ hops is:

$$P_{success}(n_i) = \prod_{m=1}^{H(n_i)} p_{(m,m+1)}$$

This expression makes explicit an important design trade-off already implicit in Section 3.3.2: while multi-hop routing reduces transmission energy, it also reduces end-to-end reliability, since $P_{success}(n_i)$ decreases monotonically with the number of hops $H(n_i)$ whenever $p_{ij} < 1$. The reliability objective of Section 3.3.3 must therefore be optimised jointly with the energy objective of Section 3.3.2, rather than in isolation — a coupling that is captured formally in the consolidated multi-objective formulation of Section 3.5.

Selection of Weighting Coefficients

The weighting coefficients w_1 and w_2 in the composite reliability objective are not fixed constants but are selected according to the operational priority of the deployment phase:

- During **routine periodic logging** (regular thermal and electrical trend recording for performance analysis of the PV/T system), delivery completeness is prioritised over speed, favouring $w_1 > w_2$.
- During **alert-triggered transmission** (an over-temperature or over-current condition that must reach the Telegram bot with minimal delay), responsiveness is prioritised, favouring $w_2 > w_1$, and is additionally subject to the hard latency bound L_{aler}^{max} defined in Section 3.4.3.

This distinction motivates the introduction of **packet priority classes** (P_{alert} and $P_{routine}$), which are formalised as part of the sampling and data-rate constraints in the following section.

4.5 Design Constraints

The objective functions defined above are subject to a set of technical constraints imposed by the physical capabilities of the NI WSN hardware platform and the requirements of the PV/T monitoring application.

4.5.1 Energy Constraints

The residual energy of every node must remain non-negative throughout the network's operational lifetime, and must not exceed the maximum battery capacity E_{max} of the NI WSN-3212/3202 node:

$$0 \le E(t) \le E_{max}, \quad \forall n_i \in V, \ \forall t \in [0, T]$$

The energy consumed between two consecutive recharge/replacement cycles must not exceed the available capacity:

$$\sum_{t=0}^{T} E_{i}^{total}(t) \le E \quad - E \ , \quad \forall n \in$$

To extend T_{life}, the duty-cycling constraint restricts the fraction of time a node remains active:

$$\delta_i(t) = \{ \ , \ node active (sensing/transmitting) 0, \ node in sleep mode \ subject to \ \frac{1}{T} \int_0^T s_i(t)\, dt \le$$

where δ_{max} is the maximum permissible duty cycle that still satisfies the minimum sampling-rate requirement of the application.

Discrete-Time Battery Discharge Model

For implementation in the discrete-time LabVIEW control loop, the continuous energy constraint above is reformulated as a recursive update equation. Letting ΔT denote the control-loop iteration period, the residual energy of node n_i at iteration $t+1$ is:

$$E(t+1) = E(t) - s(t) \cdot \left[E^{sense} + E^{proc} + E_{tx}(l_i, d_i) \right] \cdot \Delta T$$

This recursive form is directly suitable for the energy-budget bookkeeping performed within the supervisory LabVIEW VI, which maintains a running estimate of $E(t)$ for every node based

on

the known duty cycle, sampling rate, and routing distance, and raises a low-battery diagnostic event when $E(t)$ approaches E_{th}.

Operational Implication for Battery Maintenance Scheduling

The minimum operational threshold E_{th} is set conservatively above the absolute hardware cut-off voltage of the NI WSN node, in order to provide a *maintenance window* during which a technician can be notified (via the same Telegram bot channel used for sensor alerts) to replace or recharge the affected node's battery before it actually ceases to operate. This introduces a derived constraint relating the threshold margin to the maintenance response time Δ_{maint}:

$$E_{th} \geq E_i^{rate} \cdot \Delta_{maint}$$

where E_i^{rate} is the node's average discharge rate. This ensures that, even under continuous worst-case operation, the node does not reach complete depletion before the maintenance action can be carried out.

4.5.2 Communication and Range Constraints

Each NI WSN-3212/3202 node must lie within the maximum reliable transmission range $R_{\max}$ of either the gateway or a designated relay node, in accordance with the IEEE 802.15.4 radio specifications of the NI WSN platform:

$$p_i - p_{GW2} \leq R_{\max} \qquad (single - hopcase)$$

$$p_i - p_{r2} \leq R_{\max} \quad and \quad p_r - p_{GW2} \leq R_{\max} \qquad (multi - hopcase)$$

The received signal strength must additionally satisfy the minimum receiver sensitivity threshold $RSSI_{min}$ to guarantee link reliability:

$$RSSI(n_i, n_j) \geq RSSI_{min}, \quad \forall (n_i, n_j) \in E$$

The number of hops $H(n_i)$ from any node to the gateway is bounded by the maximum permissible network diameter H_{max}, beyond which latency and packet-loss probability become unacceptable for real-time alerting:

$$H(n_i) \leq H_{max}, \quad \forall n_i \in V$$

Link Budget and Path-Loss Model

The $RSSI_{min}$ constraint introduced above is, in practice, derived from a standard **link-budget equation**. For a transmit power P_{tx} (in dBm) and a log-distance path-loss model with path-loss exponent n_{PL}, the received signal strength at distance d is:

$$RSSI(d) = P_{tx} - PL(d_{ref}) - 10 n_{PL} \log_{10} \frac{d}{d_{ref}}$$

where $PL(d_{ref})$ is the measured path loss at a known reference distance d_{ref} (typically 1 m). Substituting the constraint $RSSI(d) \geq RSSI_{min}$ and solving for d yields an explicit, closed-form expression for the maximum reliable transmission range as a function of the transmit power level:

$$R_{\max}(P_{tx}) = d_{ref} \cdot 10^{\frac{P_{tx} - PL(d_{ref}) - RSSI_{min}}{10\, n_{PL}}}$$

This relation is significant for the decision-variable formulation in Section 3.5, since it shows that the transmission range $R_{\max}$ is not a single fixed hardware constant but is itself a function of the chosen transmit power level P_{tx} — which is precisely why P_{tx} is retained as a controllable decision variable rather than a fixed parameter.

Practical Deployment Implication for the PV/T Array

In the specific context of the PV/T collector geometry, the path-loss exponent n_{PL} is elevated relative to free-space propagation due to reflective interference from the metallic absorber plate and the support frame of the collector array. Consequently, the effective $R_{\max}$ achieved in practice is typically lower than the manufacturer's open-field rating for the NI WSN radio module, which motivates the conservative relay-based multi-hop topology described in Section 3.2.2 rather than relying exclusively on direct single-hop links from every node to the NI WSN-9791 gateway.

4.5.3 Sampling and Data-Rate Constraints

Each thermocouple node (τ_i = 3212) samples thermal data at a periodic interval Δt_T, while each voltage/current node (τ_i = 3202) samples electrical data at interval Δt_E. These intervals are bounded by the Nyquist requirements of the PV/T thermal time-constant and by the energy- duty-cycle constraint:

$$\Delta t_{min} \leq \Delta t_\tau \leq \Delta t_{max}, \quad \tau \in \{T, E\}$$

Justification of the Sampling Bounds via Thermal Time-Constant

The upper bound Δt_{max} is not an arbitrary design choice but is derived from the thermal response characteristics of the PV/T absorber plate. Modelling the absorber as a first-order thermal system with time constant τ_{th} (a function of the plate's thermal mass and the convective heat-transfer coefficient to the working fluid), the Nyquist–Shannon sampling criterion requires the sampling frequency $f_T = 1/\Delta t_T$ to satisfy:

$$f_T \geq \frac{2}{\tau_{th}} \quad \Rightarrow \quad \Delta t_{max} \leq \frac{\tau_{th}}{2}$$

Sampling slower than this bound risks aliasing transient thermal events — such as rapid temperature excursions caused by intermittent cloud cover — into an inaccurate or delayed representation of the true thermal profile.

The lower bound Δt_{min}, conversely, is dictated by the energy constraint of Section 3.4.1: sam- pling faster than necessary consumes battery energy without providing additional useful infor- mation once the sampling frequency already exceeds the Nyquist rate. The achievable operating point is therefore the value of Δt_T within $[\Delta t_{min}, \Delta t_{max}]$ that minimises E^{total} while still sat- isfying the Nyquist bound — in practice, Δt_T is set as close to Δt_{max} as the thermal dynamics permit.

Channel Capacity and Collision Avoidance

The aggregate data rate generated by the network must not exceed the effective channel capacity C_{ch} of the shared wireless medium, in order to avoid buffer overflow:

$$\sum_{i=1}^{k} \frac{L_i}{\Delta t_i} \leq C_{ch}$$

To prevent **packet collisions** when multiple nodes transmit concurrently — particularly during simultaneous threshold-triggered alert events — a time-division access constraint is imposed on the transmission schedule $\sigma_i(t)$ of each node:

$$\sigma_i(t) \cap \sigma_j(t) = \emptyset, \quad \forall i \neq j, \forall t$$

For the underlying CSMA/CA-based medium-access scheme employed by the IEEE 802.15.4 radio, the probability of a collision occurring when k active nodes contend for the channel within a single contention window is approximated by:

$$P_{coll}(k) = 1 - (1 - p_a)^{k-1}$$

where p_a is the per-node channel-access probability in a given slot. This expression formalises the intuitive observation that increasing the number of nodes contending simultaneously — for example, during a network-wide alert burst triggered by a sudden over-temperature event affecting multiple thermocouples at once — rapidly increases the collision probability, which is precisely the scenario the priority-class mechanism described below is designed to mitigate.

Alert Prioritisation and Latency Bound

Real-time alert packets (e.g., over-temperature or over-current events forwarded to the Telegram bot) are assigned a priority class $P_{alert} > P_{routine}$ and must satisfy a bounded worst-case latency L_{alert}:

$$L_{alert} \leq L_{aler}^{max}$$

ensuring that the notification reaches the Python-based alerting layer within an operationally acceptable time window.

In practice, this priority distinction is enforced at the LabVIEW supervisory layer by reserving a dedicated, contention-free transmission slot σ_{alert} within each superframe, distinct from the contention-based slots used for routine periodic logging traffic:

$$\sigma_{alert} \cap \sigma(t) = , \quad \forall n_i \in C_T \cup C_E$$

This guarantees that an alert packet is never forced to contend with routine traffic for channel access, directly bounding L_{alert} independently of the instantaneous network load generated by routine sampling.

4.6 Decision Variables

The optimisation problem formulated above is solved by determining the following decision variables, which collectively define the operational behaviour of the WSN:

- **Routing Path Selection** $r_i \in Path(n_i \rightarrow GW)$ — the selected next-hop or path through which node n_i forwards its sensed data to the NI WSN-9791 gateway.
- **Sleep/Wake Scheduling** $s(t) \in \{0, 1\}$ — the binary activation state of node n_i at time t, governing the duty cycle used to conserve battery energy between sensing intervals.
- **Sampling Rate** $f_i = 1/\Delta t_i$ — the frequency at which node n_i acquires thermal or electrical readings, adjusted dynamically based on PV/T thermal dynamics and residual energy.
- **Transmission Power Level** P_i^{tx} — the radio output power selected by node n_i for each transmission, chosen as the minimum level that satisfies $RSSI_{min}$ at the receiving node or gateway.
- **Alert Trigger Threshold** θ_i — the temperature, voltage, or current threshold value beyond which node n_i flags a reading as alert-class data for immediate forwarding to the Telegram notification bot.

The complete problem formulation can therefore be summarised as a constrained multi-objective optimisation:

$$\max\ (T_{life},\ R),\quad \min\ E_{network} subject to\ 0 \le E(t) \le E_{max}, p_i - p_{j2} \le R_{max}, \Delta t_{min} \le \Delta t_i \le \Delta t_{ma}$$

over the decision variables $\{r_i, s(t), f_i, P^{tx}, \theta_i\}$ for all $n_i \in V$.

4.6.1 Mapping of Decision Variables to the LabVIEW Implementation

Each abstract decision variable defined above corresponds directly to a configurable parameter within the LabVIEW host VI that supervises the NI WSN gateway, as summarised in Table 4.7. This mapping is what allows the mathematical formulation developed in this chapter to be realised as an executable control policy on the actual deployed hardware, rather than remaining a purely theoretical construct.

Table 4.7: *Mapping between abstract decision variables and their LabVIEW/NI WSN implementation.*

Decision Variable	Mathematical Symbol	LabVIEW / NI WSN Realisation
Routing path	r_i	NI WSN-9791 mesh-routing configuration; relay-node assignment table maintained by the gateway firmware
Sleep/wake schedule	$s(t)$	Node polling interval and low-power mode configuration set via the LabVIEW WSN Configuration VI
Sampling rate	$f_i = 1/\Delta t_i$	DAQmx-style channel acquisition rate configured per NI WSN-3212/3202 channel
Transmission power	F_i^{tx}	RF output power register of the IEEE 802.15.4 transceiver, exposed through the gateway's node-configuration interface
Alert threshold	θ_i	Comparison logic implemented in the LabVIEW Block Diagram (Case Structure evaluating each acquired sample against θ_i) before invoking the Python Telegram-bot script

4.7 Numerical Case Study

To ground the abstract formulation of Sections 3.2–3.5 in the concrete context of the implemented PV/T monitoring system, this section instantiates the model with representative numerical values consistent with a small-scale campus-level PV/T installation, and traces the resulting computation through each of the three objective functions.

4.7.1 Deployment Configuration

Consider a representative deployment consisting of $k = 8$ sensor nodes: six NI WSN-3212 thermocouple nodes distributed across the absorber plate ($|\mathcal{C}_T| = 6$) and two NI WSN-3202 nodes monitoring the voltage and current of two PV strings ($|\mathcal{C}_E| = 2$), all reporting to a single NI WSN-9791 gateway. Table 4.8 lists the assumed parameter values.

Table 4.8: *Assumed parameter values for the numerical case study.*

Parameter	Symbol	Assumed Value
Battery capacity	E_{max}	3.6 V, 2400 mAh (Li-ion pack)
Energy threshold	E_{th}	10% of E_{max}
Maximum transmission range	$R_{\max}$	30 m (open-field), ≈18 m (on-array, derated)
Thermal sampling interval	Δt_T	5 s
Electrical sampling interval	Δt_E	2 s
Packet size	l	64 bytes
Channel capacity	C_{ch}	250 kbps (IEEE 802.15.4 nominal)
Maximum hop count	H_{max}	3
Alert latency bound	L_{aler}^{max}	2 s

4.7.2 Aggregate Data-Rate Verification

Applying the channel-capacity constraint of Section 3.4.3 to this configuration, the aggregate bit rate demanded by the network is:

$$\sum_{i=1}^{8} \frac{l_i}{\Delta t_i} = 6 \times \frac{64 \times 8}{5} + 2 \times \frac{64 \times 8}{2} = 6 \times 102.4 + 2 \times 256 = 614.4 + 512 = 1126.4 \text{ bps}$$

Since $1126.4 \text{ bps} \ll C_{ch} = 250{,}000 \text{ bps}$, the routine sampling traffic occupies less than 0.5% of the nominal channel capacity under normal operating conditions, confirming that the channel-capacity constraint of Section 3.4.3 is comfortably satisfied for routine logging traffic at this node density. This large margin is intentional: it preserves channel headroom for burst alert traffic and for the increased channel contention that occurs during simultaneous multi-node alert events, as analysed via the collision-probability expression of Section 3.4.3.1.

4.7.3 Energy Budget Verification

Using the first-order radio model parameters introduced in Section 3.3.2.2 and assuming an average single-hop transmission distance of $d = 8$ m (within the free-space regime, $d < d_0$), the per-transmission energy cost for a thermocouple node is:

$$E_{tx}(512\ bits, 8\ m) = 512\times50\times10^{-9}+512\times10\times10^{-12}\times8^2 \approx 2.56\times10^{-5}+3.28\times10^{-7} \approx 2.59\times10^{-5}\ J$$

Over a 24-hour monitoring horizon with $\Delta t_T = 5$ s, the number of transmissions per node is $N_{tx} = 86400/5 = 17280$, giving a daily transmission energy of:

$$E^{tx,\,daily}, = N_{tx} \times E_{tx} \approx 17280 \times 2.59 \times 10^{-5} \approx 0.448\ J/day$$

Comparing this to the usable energy of the assumed 2400 mAh Li-ion pack at 3.6 V ($E_{max} \approx 3.6\times 2400\times 3.6 \approx 31{,}104$ J, of which 90% — i.e., down to E_{th} — is usable, giving $\approx 27{,}994$ J), the transmission energy alone would, in isolation, permit:

$$T^{tx-only}_{life} \approx \frac{27{,}994}{0.448} \approx 62{,}486\ days \approx 171\ years$$

This figure is, of course, an idealised upper bound that considers transmission energy in isolation; in practice T_{life} is dominated not by E_{tx} but by the sensing energy E^{sense}, the standby (quiescent) current drawn even while the node is nominally asleep, and — critically — by environmental factors such as ambient temperature extremes that reduce the effective capacity of the Li-ion cell. The case study nonetheless demonstrates the central design insight of this chapter: for the NI WSN hardware platform at typical PV/T deployment densities, *radio transmission energy is not the binding constraint on network lifetime*; the duty-cycling policy $s(t)$ and the baseline quiescent power draw of the node electronics dominate the energy budget, which is precisely why Section 3.4.1 treats δ_{max} (duty cycle), rather than transmission distance alone, as the principal lever for extending T_{life}.

4.8 Sensitivity Analysis and Design Trade-offs

The objective functions and constraints formulated in this chapter are not independent; tightening one frequently relaxes or violates another. This section discusses the principal trade-offs that govern the final parameter selection for the deployed system, and explains how the formulation guides the resolution of each conflict.

4.8.1 Energy Lifetime versus Sampling Resolution

Reducing the sampling interval Δt_T improves the temporal resolution of the thermal monitoring (Section 3.4.3.1) but directly increases the number of sensing and transmission events per unit time, and therefore reduces T_{life} according to the recursive discharge model of Section 3.4.1.1. Formally, differentiating the daily energy expenditure with respect to the sampling interval shows an inverse relationship:

$$\frac{\partial E_i^{daily}}{\partial \Delta t_T} < 0$$

meaning that any reduction in Δt_T (faster sampling) strictly increases daily energy consumption. The Nyquist bound derived in Section 3.4.3.1 therefore acts as the natural lower limit on Δt_T : sampling faster than $\tau_{th}/2$ provides no additional information about the absorber's ther- mal state while continuing to consume battery energy, making it a strictly dominated choice in the optimisation sense.

4.8.2 Network Lifetime versus Reliability

As shown in Section 3.3.3.1, increasing the hop count $H(n_i)$ reduces per-hop transmission distance and therefore reduces E_{tx} (favouring T_{life}), but simultaneously reduces the end-to-end success probability $P_{success}(n_i)$ because of the multiplicative effect of per-hop reliability across the path. This trade-off is captured by considering the two extremes:

- **Single-hop topology** ($H(n_i) = 1$ for all n_i): maximises $P_{success}(n_i)$ for each individual

transmission (only one link to fail), but forces nodes far from the gateway to transmit at high power over long distances, incurring the d^4 multi-path energy penalty of Section 3.3.2 and disproportionately depleting the batteries of the most distant nodes — creating an unbalanced energy profile across V that accelerates the FND event.

- **Maximum-hop topology** (every node relays through the maximum permitted H_{max} hops): minimises the per-hop energy cost for distant nodes, but compounds the link-failure probability multiplicatively across H_{max} links, and concentrates relay-forwarding energy burden on the small subset of nodes nearest the gateway, which then become the new bottleneck for T_{life}.

The chosen design — a bounded multi-hop topology with $H_{max} = 3$, as instantiated in the case study of Section 3.6.1 — represents a deliberate intermediate point on this trade-off curve, selected because it keeps $P_{success}(n_i)$ above an acceptable operational floor for all nodes while still avoiding the worst-case energy imbalance of the pure single-hop topology.

4.8.3 Alert Responsiveness versus Channel Utilisation

The reservation of a contention-free alert slot σ_{alert} (Section 3.4.3.3) guarantees the latency bound L_{max}^{aler} independently of routine traffic load, but this guarantee is purchased at the cost of channel capacity that cannot be used for routine logging traffic during the reserved slot. Because Section 3.6.2 demonstrates that routine traffic occupies less than 0.5% of the available channel capacity at the case-study node density, this trade-off is judged acceptable: the opportunity cost of reserving a dedicated alert slot is negligible relative to the large unused capacity margin, while the benefit — a deterministic upper bound on alert latency, decoupled from the instantaneous collision probability $P_{coll}(k)$ of Section 3.4.3.2 — is operationally critical for a system whose explicit purpose is real-time safety alerting via the Telegram bot.

4.8.4 Summary of Trade-off Resolutions

Table 4.9 consolidates the trade-off resolutions adopted in the final system design, cross-referencing each decision back to the governing constraint or objective function defined earlier in this chapter.

Table 4.9: *Summary of design trade-off resolutions adopted for the PV/T WSN.*

Trade-off	Resolution Adopted	Governing Reference
Sampling rate vs. energy bound	Δt_T set at Nyquist $\tau_{th}/2$	Section 3.4.3.1
Hop count vs. reliability	Bounded multi-hop, $H_{max} = 3$	Section 3.2.2, 3.7.2
Alert latency vs. throughput	Dedicated contention-free alert slot	Section 3.4.3.3
Node density vs. gate-way load	$\lvert\mathcal{C}_T\rvert \gg \lvert\mathcal{C}_E\rvert$, capped by k_{max}	Section 3.2.3, 3.2.4

4.9 Notation Summary

Table 4.10: *Summary of notation used in the WSN problem formulation.*

Symbol	Definition
$G = (V, E)$	Network topology graph; nodes V and feasible communication links E
n_i, GW	Sensor node i; the NI WSN-9791 gateway node
p_i	Spatial coordinates of node n_i
$E_i(t)$	Residual battery energy of node n_i at time t
$s_i(t)$	Binary active/sleep state of node n_i
τ_i	Node hardware type (3212 thermocouple or 3202 voltage/current)
C_T, C_E	Thermal and electrical sensor clusters
A, L	Adjacency matrix and graph Laplacian of G
$\lambda_2(L)$	Algebraic connectivity of the network graph
$P_{success}(n_i)$	End-to-end path success probability for node n_i
E_{max}, E_{th}	Maximum battery capacity and minimum operational energy threshold
δ_{max}	Maximum permissible duty cycle
R_{max}, $RSSI_{min}$	Maximum transmission range and minimum receiver sensitivity
$H(n_i)$, H_{max}	Hop count of node n_i; maximum permitted hop count
τ_{th}	Thermal time constant of the PV/T absorber plate
C_{ch}	Effective wireless channel capacity
$\sigma_i(t)$, σ_{alert}	Transmission schedule of node n_i; reserved alert slot

$P_{coll}(k)$	Probability of channel-access collision among k contending nodes
L_{alert}, L^{max}	Alert packet latency and its required upper bound

Chapter 5

Project Design and Implementation

Introduction

This chapter presents the practical design and implementation of the proposed **Wireless Sensor Network (WSN)** system. It covers the configuration of the National Instruments (NI) WSN hardware platform using LabVIEW, the development of the monitoring Virtual Instrument (VI), the experimental range and link-quality testing carried out under different conditions, the integration of a second sensor node, and the implementation of a Telegram-based high-temperature alert mechanism.

5.1 LabVIEW Project Creation and Target Configuration

The implementation begins by creating a dedicated LabVIEW project to host the Wireless Sensor Network targets, nodes, and Virtual Instruments (VIs). This project, named **WSN_Building.lvproj**, was created using the LabVIEW 2013 Real-Time Module, which provides native support for NI Wireless Sensor Network (NI-WSN) hardware.

5.1.1 Project Creation

From the LabVIEW startup screen, a new project is created by selecting **Create Project** and then choosing the **Blank Project** template, which provides an empty project structure that can be populated with targets, devices, and VIs as required by the application.

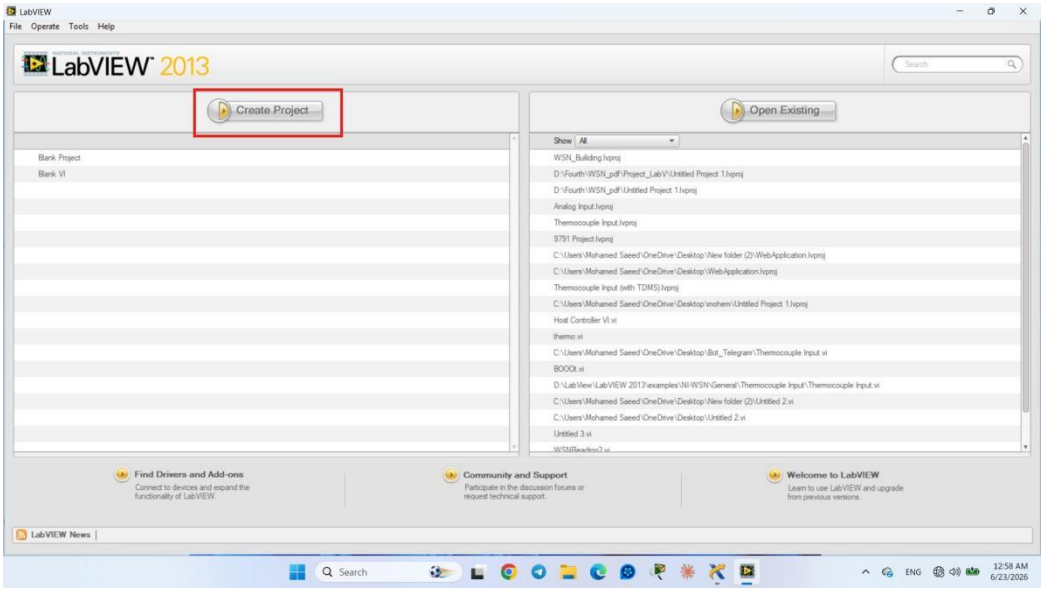

Figure 5.63: *LabVIEW 2013 startup screen showing the Create Project option.*

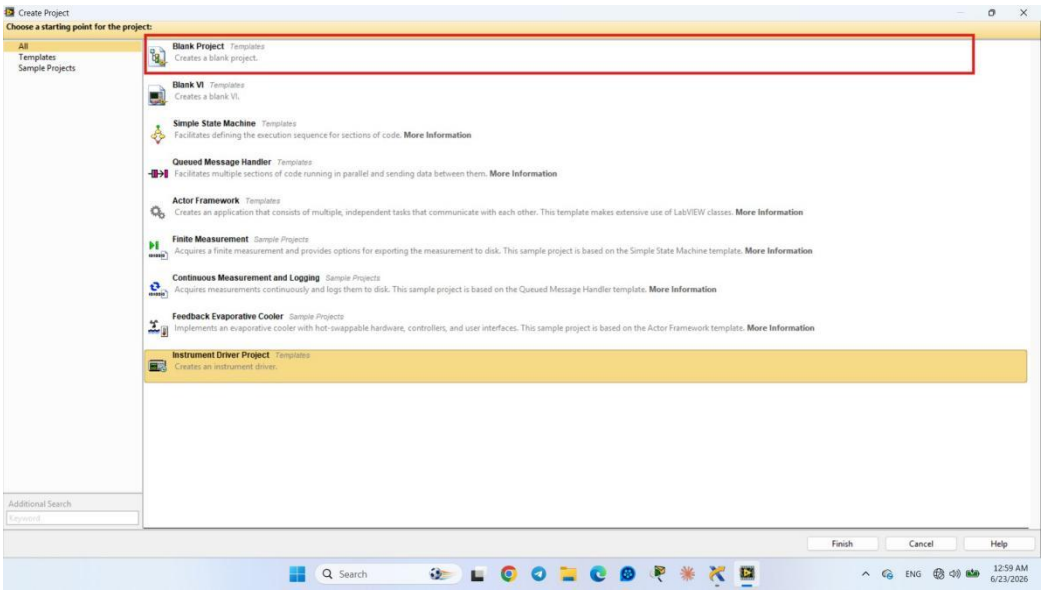
Create Project
Choose a starting point for the project:
All
Templates
Sample Projects
Blank Project
Creates a blank project.
Blank VI
Creates a blank VI.
Simple State Machine
Facilitates defining the execution sequence for sections of code. More Information
Queued Message Handler
Facilitates multiple sections of code running in parallel and sending data between them. More Information
Actor Framework
Creates an application that consists of multiple, independent tasks that communicate with each other. This template makes extensive use of LabVIEW classes. More Information
Finite Measurement
Acquires a finite measurement and provides options for exporting the measurement to disk. This sample project is based on the Simple State Machine template. More Information
Continuous Measurement and Logging
Acquires measurements continuously and logs them to disk. This sample project is based on the Queued Message Handler template. More Information
Feedback Evaporative Cooler
Implements an evaporative cooler with hot-swappable hardware, controllers, and user interfaces. This sample project is based on the Actor Framework template. More Information
Instrument Driver Project
Creates an instrument driver.
Additional Search
Finish
Cancel
Help
Search
ENG
12:59 AM
6/23/2026

Figure 5.64: *Selecting the Blank Project template to begin a new LabVIEW project.*

Once created, the project opens in the **Project Explorer** window, showing the default project tree consisting of *My Computer*, *Dependencies*, and *Build Specifications*.

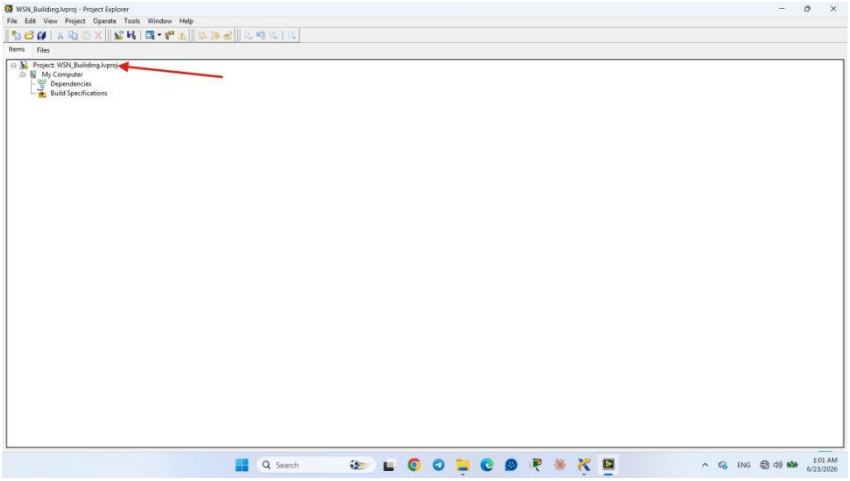

Figure 5.65: *Newly created WSN_Building.lvproj shown in the Project Explorer.*

5.1.2 Adding the WSN Gateway and Node

The next step is to add the physical WSN hardware to the project. By right-clicking *My Computer* and selecting **New → Targets and Devices...**, LabVIEW scans the network for available NI-WSN gateways and end nodes.

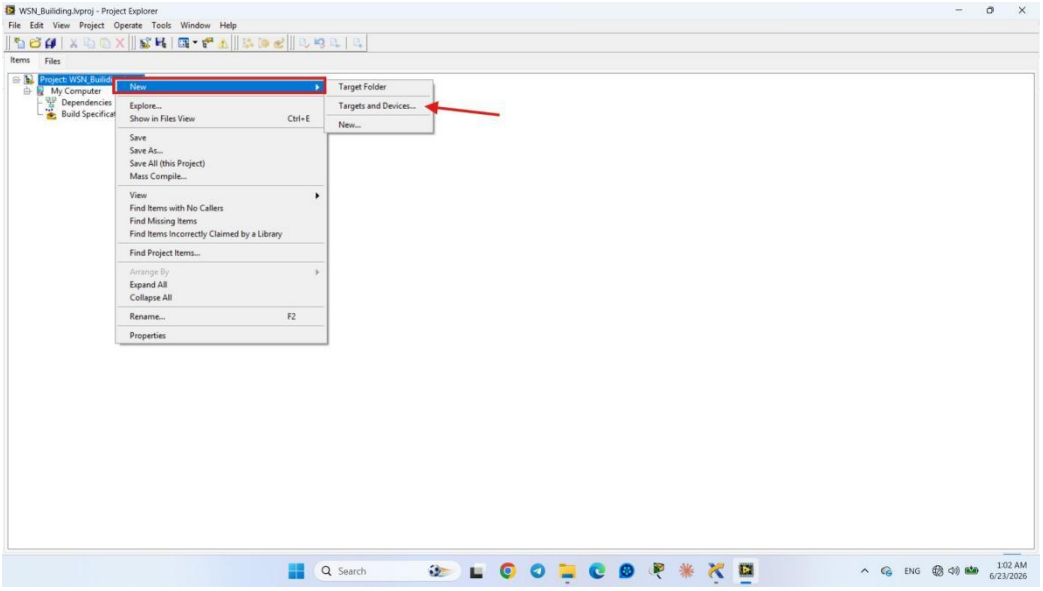

Figure 5.66: *Adding a new target or device to the LabVIEW project.*

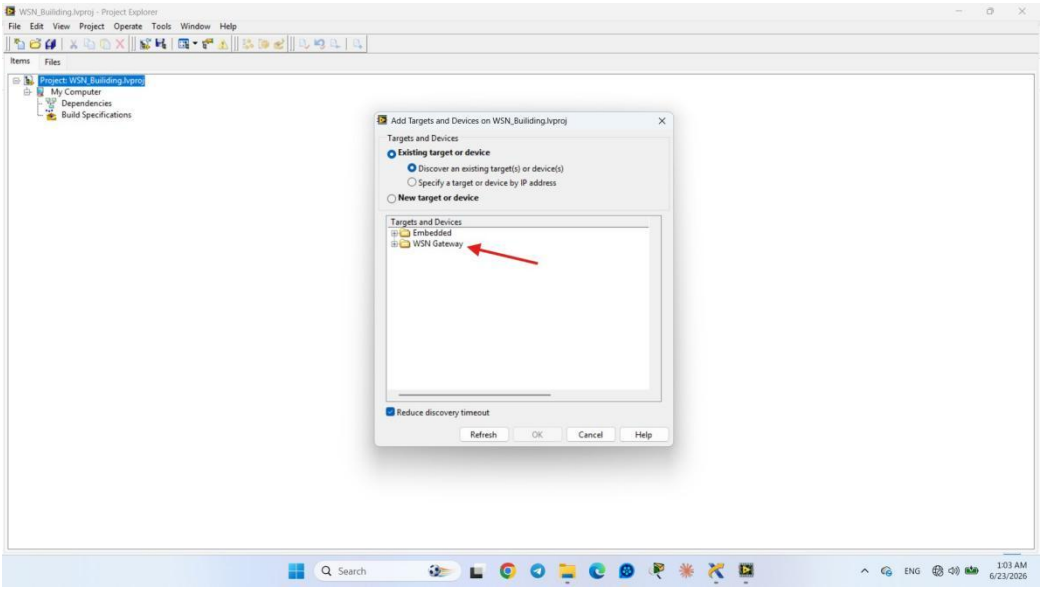

Figure 5.67: *Discovering existing targets and devices, including the WSN Gateway.*

After discovery, the **NI-WSN9791** gateway is added to the project at IP address `169.254.11.67`, together with its associated *Wireless Sensor Network* folder and the first end node (**Node2**, NI WSN-3212, Serial Number 199F09A, ID 2).

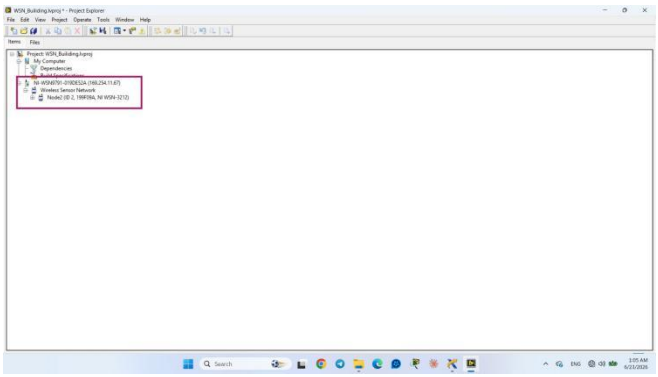

Figure 5.68: *Project tree after adding the WSN Gateway and the first sensor node.*

5.1.3 Deploying the Gateway and Node

Before the gateway and node variables can be used in a VI, they must be **deployed** so that LabVIEW establishes communication with the physical hardware. Deployment is performed by right-clicking the gateway, and then the node, and selecting **Deploy** (or **Deploy All**) from the context menu.

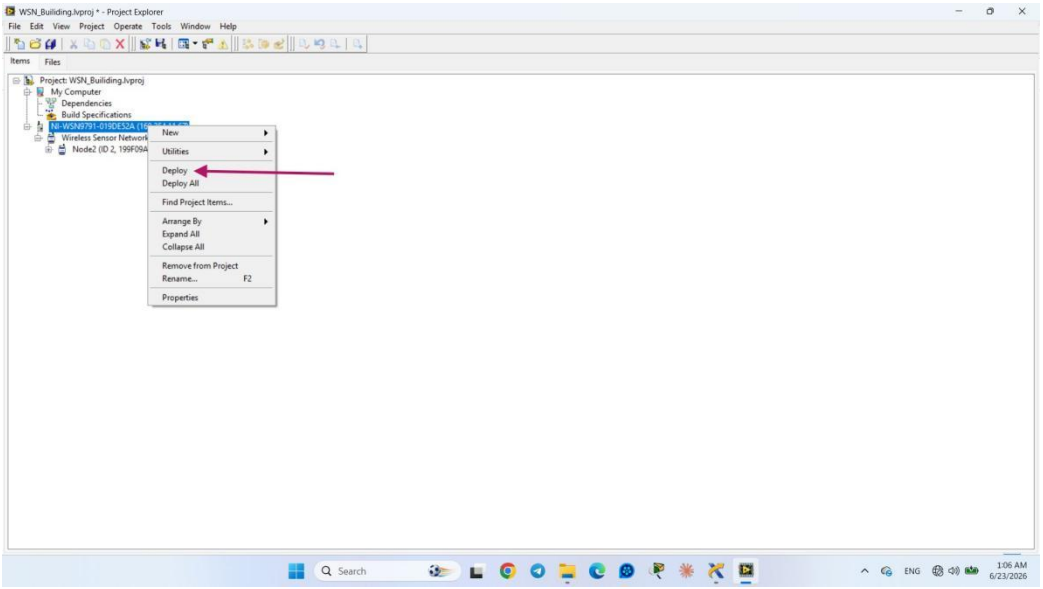

Figure 5.69: *Deploying the NI-WSN9791 gateway to establish communication with the hardware.*

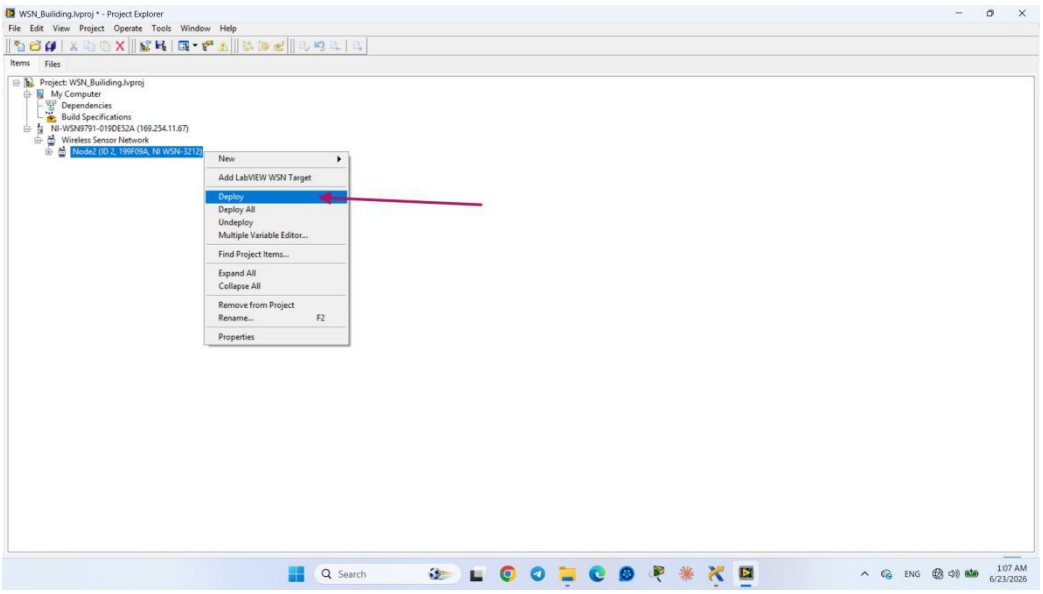

Figure 5.70: *Deploying the wireless sensor node (Node2, NI WSN-3212).*

Once deployed, the node automatically exposes its available I/O variables in the project tree, including *Battery Voltage*, *Link Quality*, *Mesh Router*, four digital I/O channels (*DIO0–DIO3*), and four thermocouple input channels (*TC0–TC3*), in addition to *External Power* status.

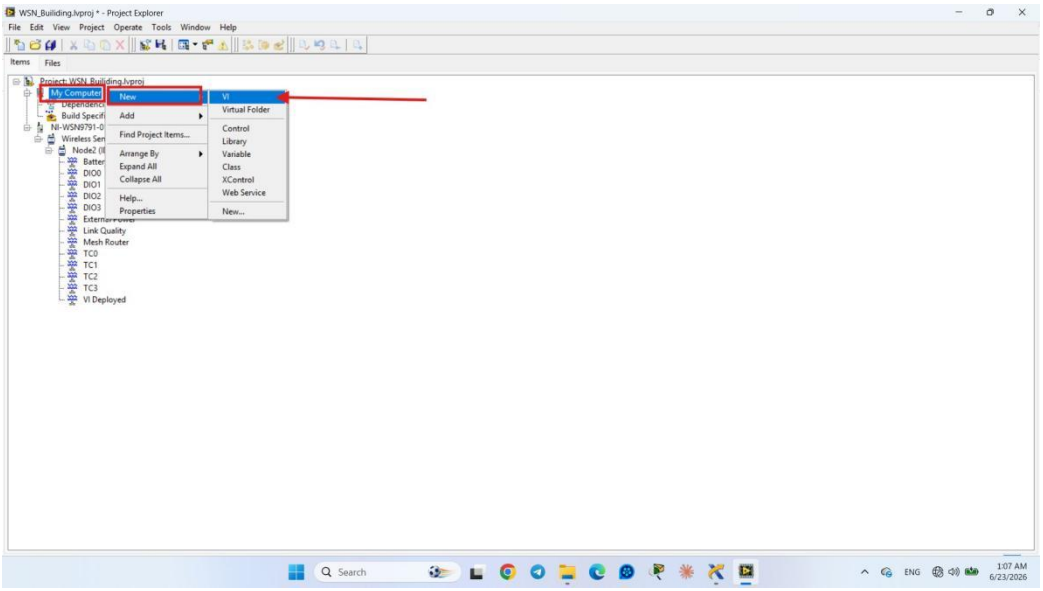

Figure 5.71: *Creating a new VI from the project and the exposed node variables (Battery Voltage, Link Quality, DIO0–DIO3, TC0–TC3, Mesh Router).*

5.1.4 Verifying the Node in Measurement & Automation Explorer

The deployed gateway and node can also be verified independently of LabVIEW using NI's **Measurement & Automation Explorer (MAX)**. Under *Remote Systems*, the gateway

`NI-WSN9791-019DE52A` lists the connected wireless node, its serial number, node ID, battery state, link quality, network mode, and firmware version.

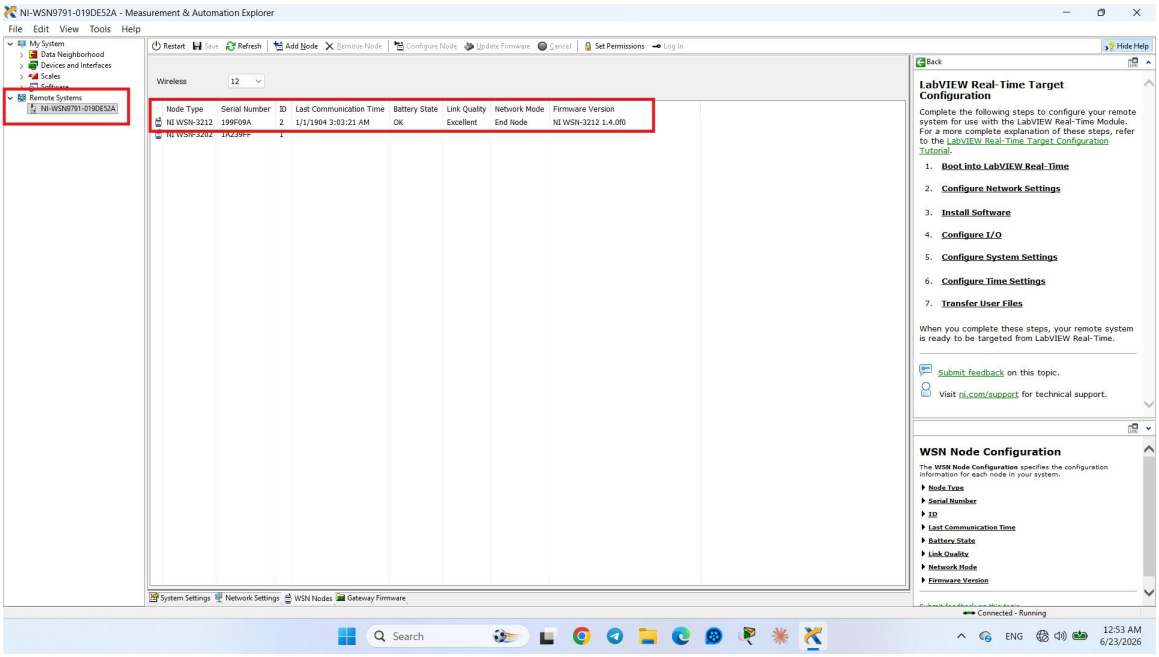
NI-WSN9791-019DE52A - Measurement & Automation Explorer
File Edit View Tools Help
My System
Data Neighborhood
Devices and Interfaces
Scales
Remote Systems
NI-WSN9791-019DE52A
Restart Save Refresh Add Node Remove Node Configure Node Update Firmware Cancel Set Permissions Log In
Hide Help
Wireless 12
Node Type Serial Number ID Last Communication Time Battery State Link Quality Network Mode Firmware Version
NI WSN-3212 199F09A 2 1/1/1904 3:03:21 AM OK Excellent End Node NI WSN-3212 1.4.0f0
NI WSN-3202 1A239FF 1
System Settings Network Settings WSN Nodes Gateway Firmware
Back
LabVIEW Real-Time Target Configuration
Complete the following steps to configure your remote system for use with the LabVIEW Real-Time Module. For a more complete explanation of these steps, refer to the LabVIEW Real-Time Target Configuration Tutorial.
1. Boot into LabVIEW Real-Time
2. Configure Network Settings
3. Install Software
4. Configure I/O
5. Configure System Settings
6. Configure Time Settings
7. Transfer User Files
When you complete these steps, your remote system is ready to be targeted from LabVIEW Real-Time.
Submit feedback on this topic.
Visit ni.com/support for technical support.
WSN Node Configuration
The WSN Node Configuration specifies the configuration information for each node in your system.
Node Type
Serial Number
ID
Last Communication Time
Battery State
Link Quality
Network Mode
Firmware Version
Connected - Running
Search
ENG
12:53 AM
6/23/2026

Figure 5.72: *Measurement & Automation Explorer (MAX) showing the connected NI WSN-3212 node with Excellent link quality and OK battery state.*

5.2 Monitoring Virtual Instrument (VI) Development

After successfully deploying the WSN gateway and node, a dedicated VI named **Building_WSN.vi** was developed to continuously monitor the key parameters reported by the wireless node: *Battery Voltage*, *Link Quality*, and the *Thermocouple (TC0)* temperature reading.

5.2.1 Block Diagram

In the block diagram, the three WSN shared variables — *Battery Voltage*, *Link Quality*, and *TC0* — are read inside a **While Loop** that executes once every second, controlled by a *Wait* function with a 1000 ms timer. The TC0 value is simultaneously routed to a numeric indicator and to a graphical thermometer display. A *stop* Boolean control is used to terminate the loop.

5.2.2 Front Panel

The front panel presents the live readings in a simple, readable layout: a graphical **Thermometer** for the temperature reading, together with numeric indicators for *Battery Voltage 2*, *Link*

Quality, and *TC0*.

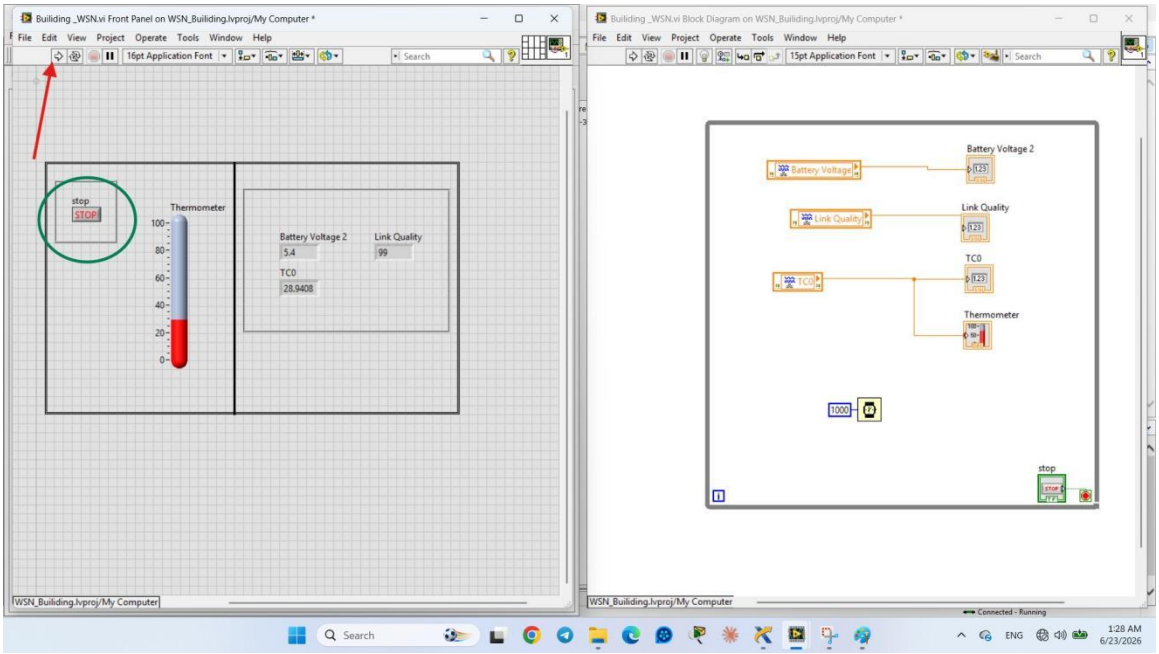

Figure 5.73: *Front panel and block diagram of Building_WSN.vi showing real-time Battery Voltage, Link Quality, and TC0 temperature readings.*

5.3 Experimental Range and Link Quality Testing

To evaluate the wireless performance of the NI WSN-3212 node, a series of field tests were performed by varying the physical distance between the sensor node and the gateway while observing the resulting *Link Quality* value reported on the front panel. The node was powered from a portable battery pack and carried to different locations, including an outdoor staircase and a multi-floor indoor stairwell, to emulate realistic deployment conditions with and without obstructing walls.

Figure 5.74: *Field test setup: the WSN node and thermocouple probe carried outdoors during the range test.*

5.3.1 Test Case 1: Short Range (20 cm)

At a very short distance of approximately 20 cm between the node and the gateway, the system reported a near-perfect **Link Quality of 99**, confirming reliable communication at close range.

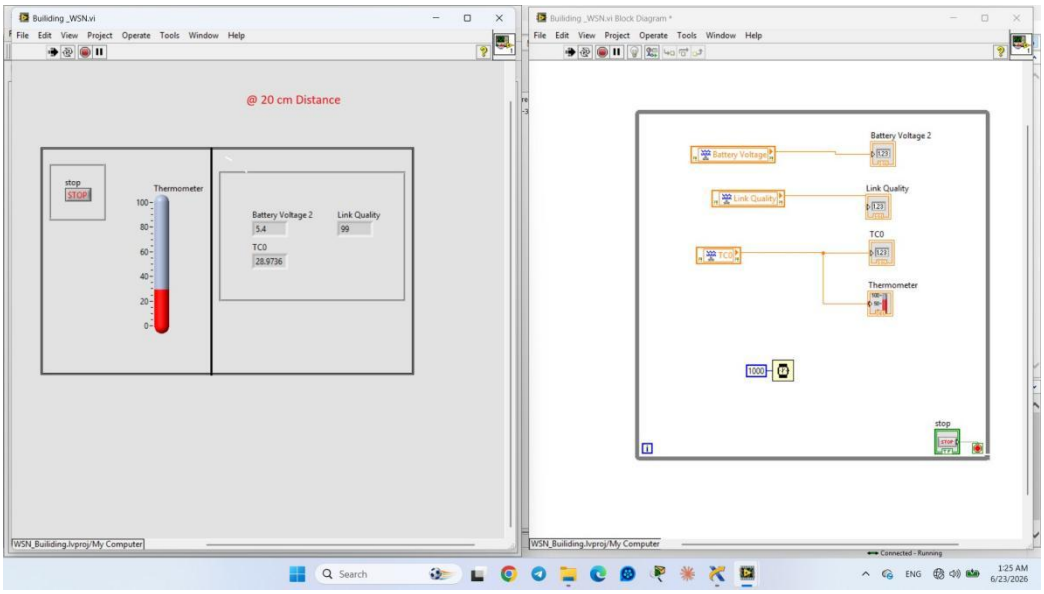

Figure 5.75: *Link Quality of 99 recorded at a distance of approximately 20 cm.*

5.3.2 Test Case 2: Distance Greater Than 20 m

When the node was moved to a distance exceeding 20 m from the gateway, within the same building stairwell, the **Link Quality dropped to 50**, indicating a noticeable but still acceptable degradation in signal strength.

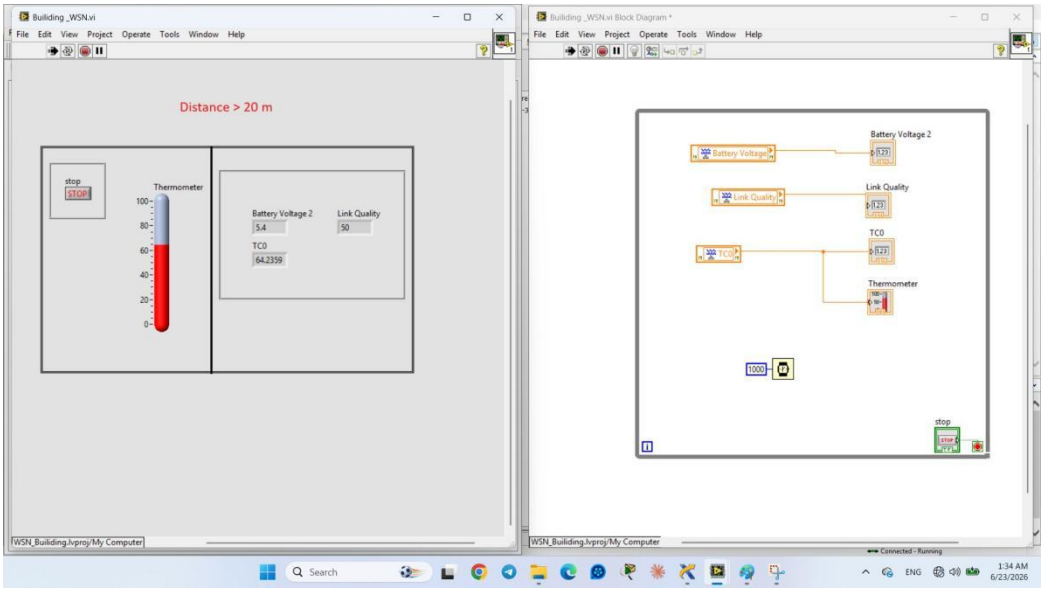

Figure 5.76: *Link Quality of 50 recorded at a distance greater than 20 m.*

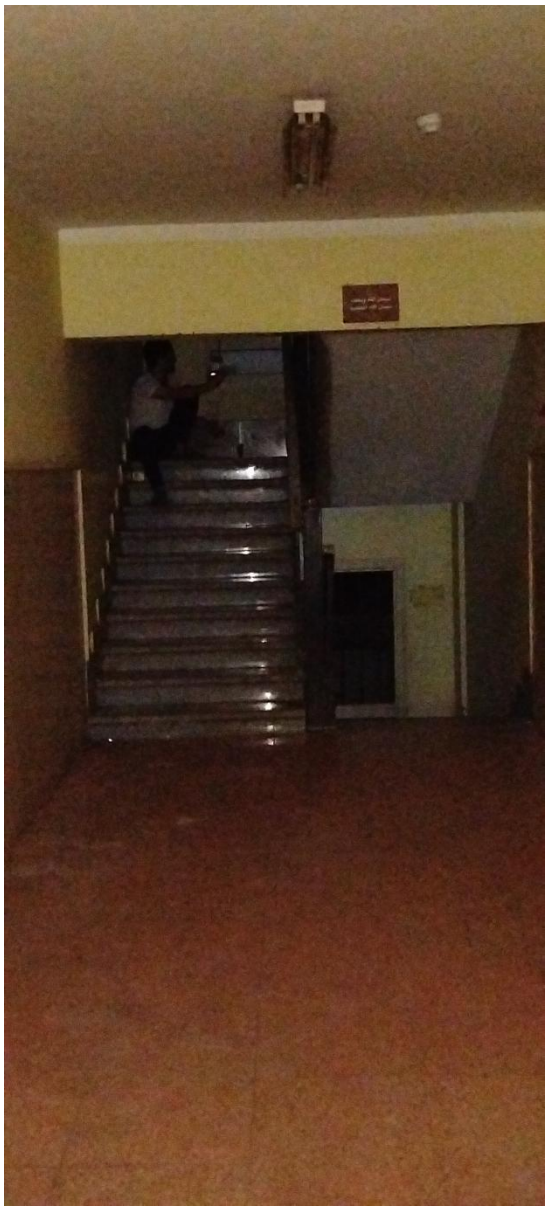

Figure 5.77: *Multi-floor indoor stairwell used for the long-distance link quality test.*

5.3.3 Test Case 3: Distance Greater Than 40 m, With Walls

Increasing the separation to over 40 m and introducing several concrete walls between the node and the gateway caused a further drop in signal quality, with the **Link Quality falling to 22**, demonstrating the attenuating effect of physical obstructions on the wireless link.

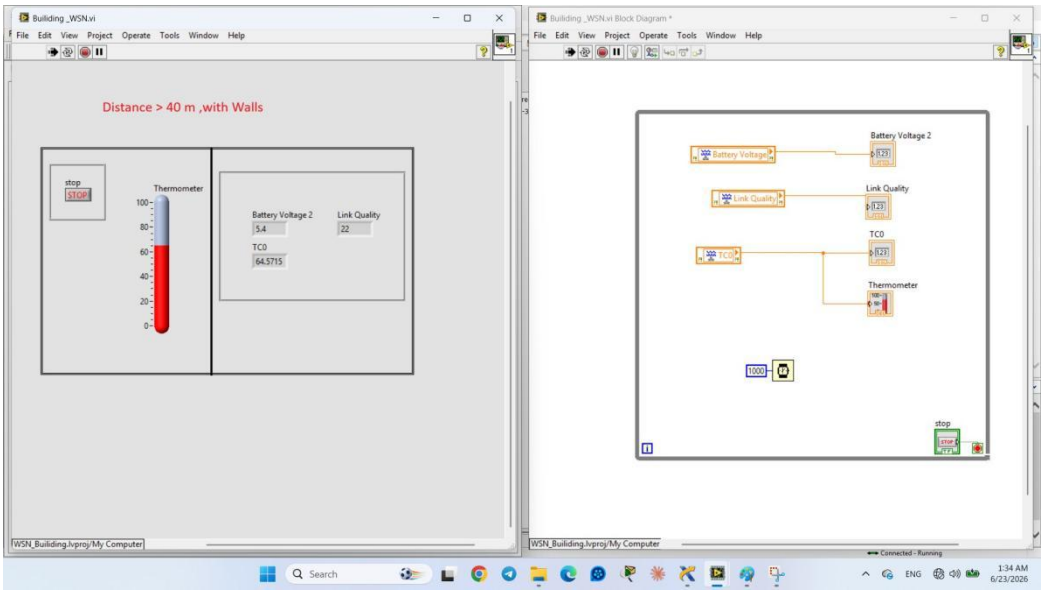

Figure 5.78: *Link Quality of 22 recorded at a distance greater than 40 m with intervening walls.*

5.3.4 Test Case 4: Distance Greater Than 50 m

At the maximum tested distance of over 50 m, the **Link Quality decreased further to 17**, representing the practical communication range limit observed for the NI WSN-3212 node under the tested conditions.

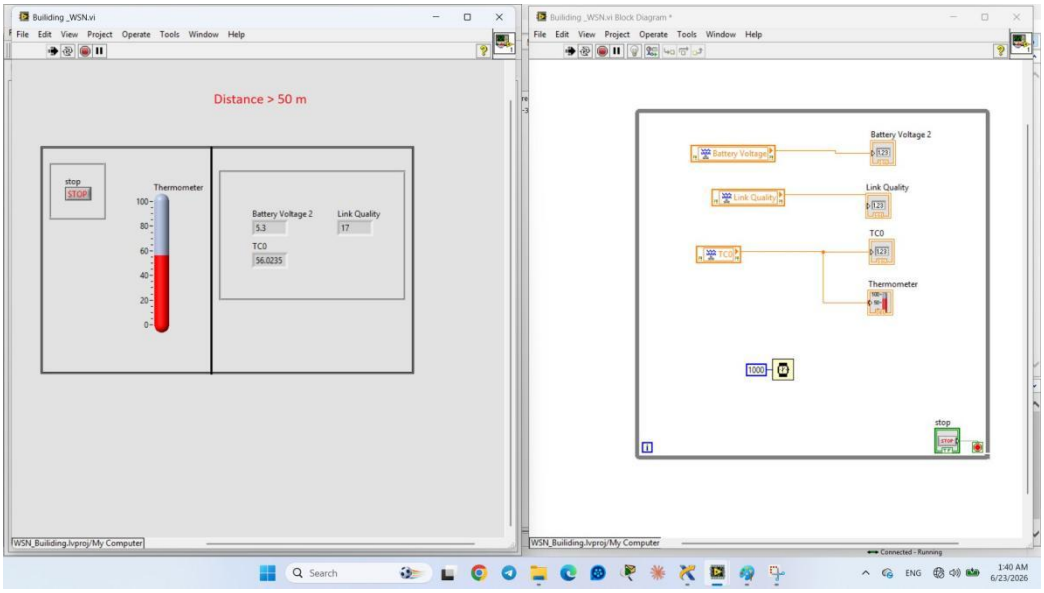

Figure 5.79: *Link Quality of 17 recorded at a distance greater than 50 m.*

5.3.5 Summary of Range Test Results

Table 5.11 summarizes the link quality measurements obtained at each tested distance, clearly showing the expected inverse relationship between distance (and physical obstructions) and

wireless link quality.

Table 5.11: *Summary of distance versus link quality test results.*

Test Case	Condition	Link Quality
1	20 cm (line of sight)	99
2	> 20 m (indoor stairwell)	50
3	> 40 m, with walls	22
4	> 50 m	17

5.4 Multi-Node System: Thermocouple and Soil Moisture Sensors

To extend the system beyond a single sensing point, a second NI WSN node (**NI WSN-3202**, Serial Number 1A239FF, ID 1) was added to the project and deployed alongside the original NI WSN-3212 node. Both nodes were verified in MAX, where each reported *Excellent* link quality and an *OK* battery state.

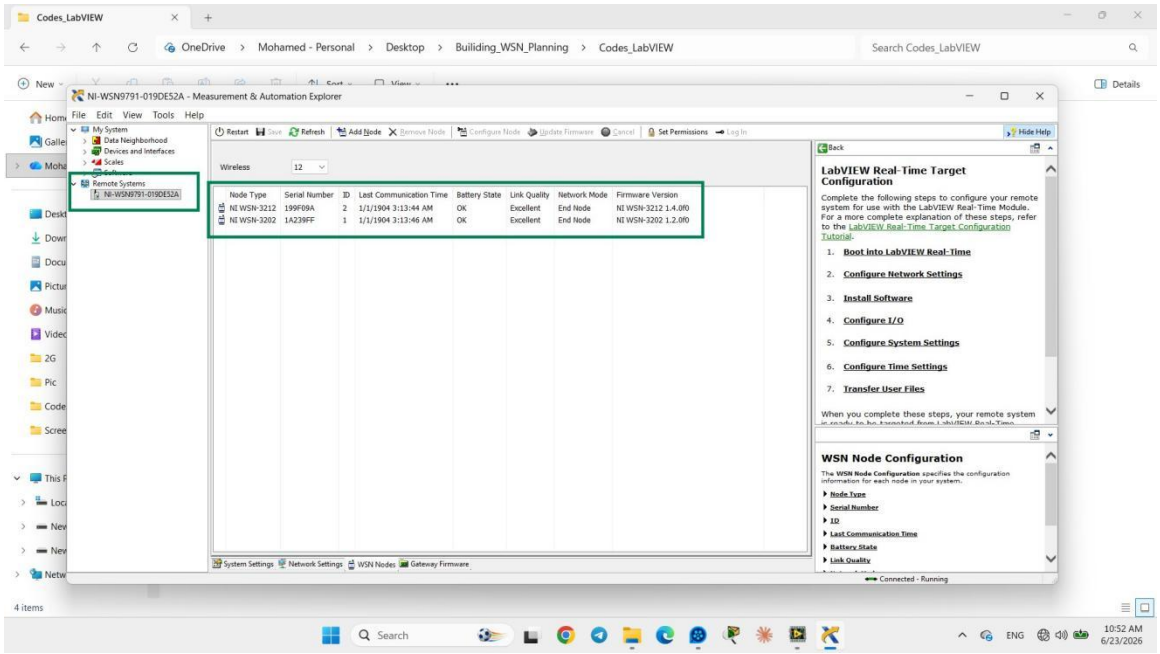

Figure 5.80: *Measurement & Automation Explorer showing both deployed nodes (NI WSN-3212 and NI WSN-3202) with Excellent link quality.*

5.4.1 Sensor Assignment

The two nodes were assigned to two different sensing functions:

- **Node 1 (NI WSN-3212)** — equipped with a **thermocouple sensor**, reading two temperature channels, *TC0* and *TC1*, in addition to its *Battery Voltage* and *Link Quality*.
- **Node 2 (NI WSN-3202)** — equipped with a **soil moisture sensor** connected to its analog input channel (*AI0*), in addition to its own *Battery Voltage 2* and *Link Quality 2*.

5.4.2 Updated Block Diagram

The block diagram of `Building_WSN.vi` was extended to read both nodes' variables within the same timed loop. The thermocouple channels (*TC0*, *TC1*) drive two thermometer-style indicators (*Sensor_1* and *Sensor_2*), while the soil-moisture analog input (*AI0*) is displayed numerically and plotted on a **Waveform Chart**.

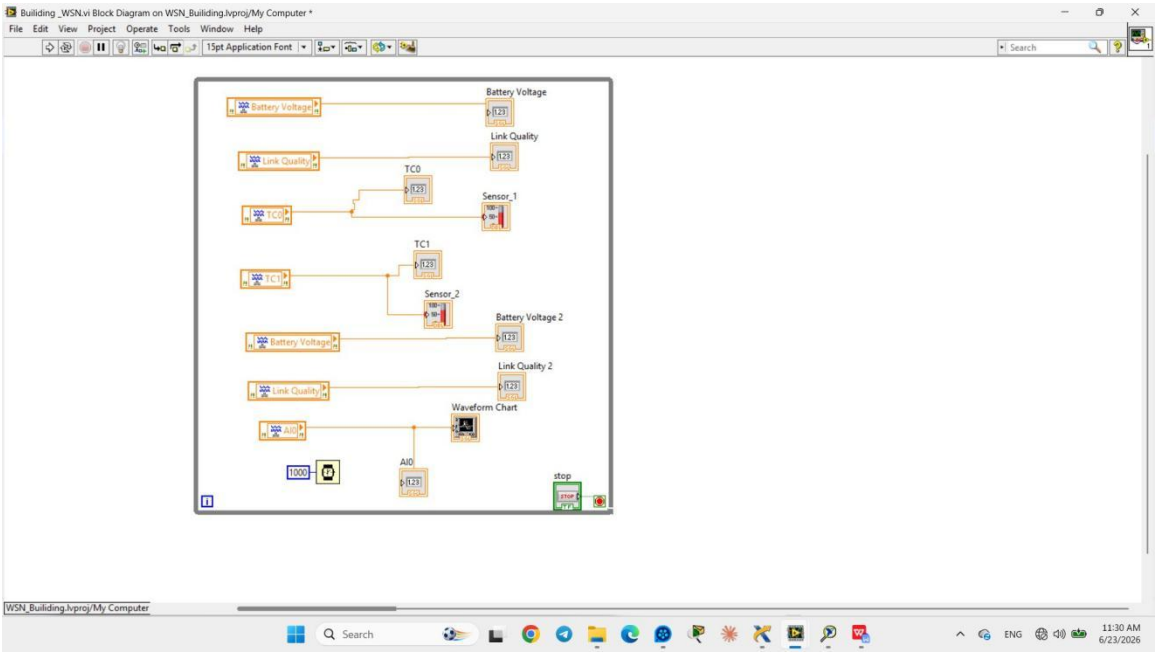

Figure 5.81: *Block diagram reading Battery Voltage, Link Quality, TC0, TC1, Battery Voltage 2, Link Quality 2, and AI0 from both nodes.*

5.4.3 Updated Front Panel

The front panel was reorganized into two clearly labeled sections, one per node, each showing a photo-style representation of the WSN node together with its corresponding readings and indicators.

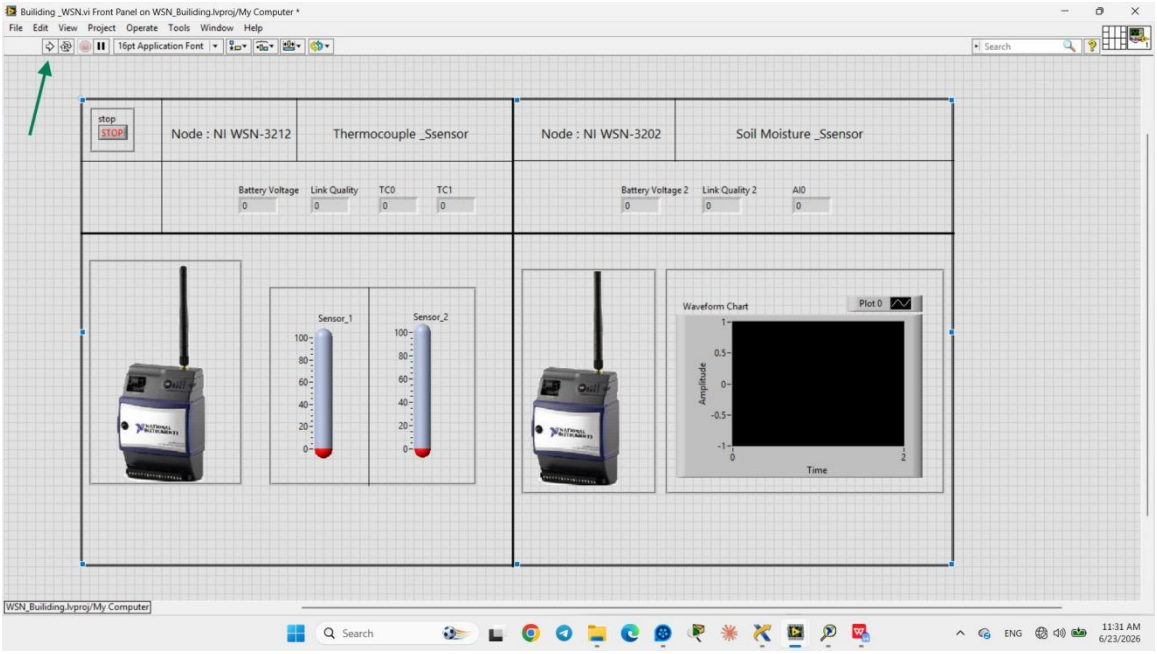

Figure 5.82: *Front panel layout for the two-node system: Node NI WSN-3212 (Thermocouple Sensor) and Node NI WSN-3202 (Soil Moisture Sensor).*

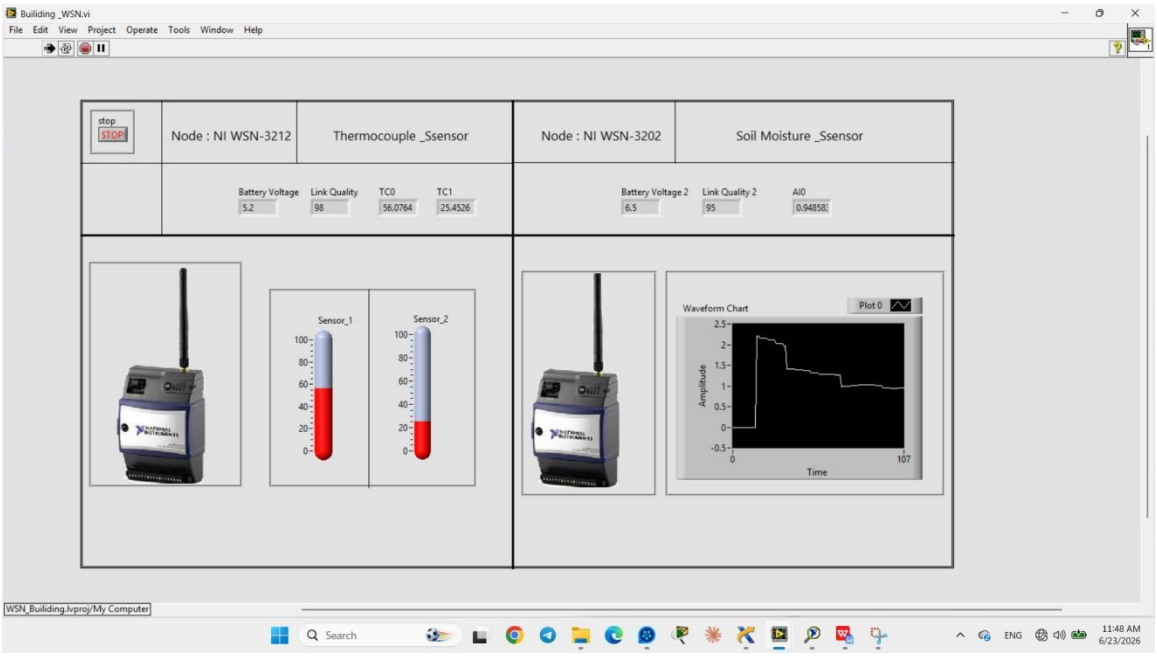

Figure 5.83: *Live operation of the two-node system showing TC0 = 56.08 °C, TC1 = 25.45 °C, and a soil moisture reading of AI0 = 0.949, together with the corresponding waveform chart.*

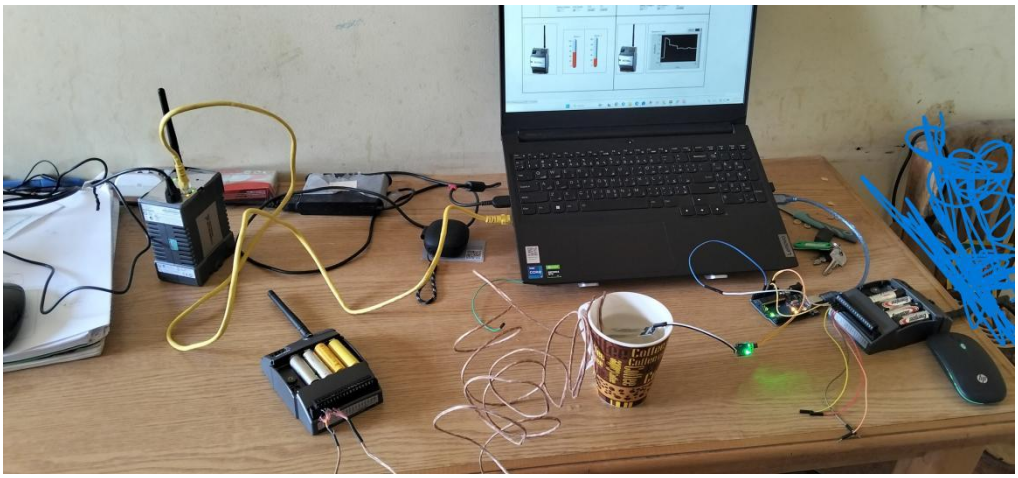

Figure 5.84: *Laboratory setup of the complete two-node WSN system, including the gateway, both sensor nodes, the soil moisture sensor board, the thermocouple probe, and the host laptop running LabVIEW.*

5.5 Host Controller VI and High-Temperature Alert System

In the final implementation stage, a dedicated **Host Controller VI.vi** was developed to serve as the main supervisory interface of the system. This VI integrates the live sensor readings with a visual high-temperature alarm and an automated **Telegram** notification mechanism.

5.5.1 Block Diagram and Alarm Logic

The Host Controller VI reads the *Link Quality*, *Battery Voltage*, and *TC0* variables from the deployed WSN node and displays them on numeric indicators, a graphical thermometer, and a waveform chart. A comparison block checks whether the TC0 reading exceeds a threshold of **40 °C**; the Boolean result of this comparison drives both the *High Temp Alarm* indicator on the front panel and the execution of an external Python script, `telegram_alert.py`, via a **System Exec** call.

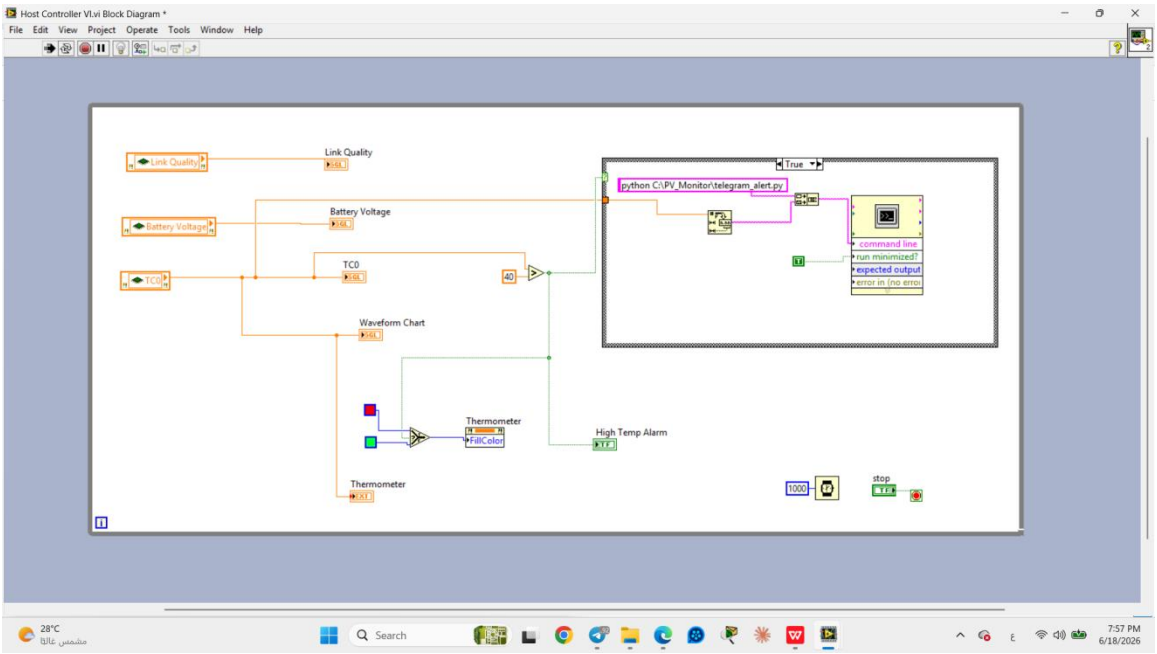

Figure 5.85: *Block diagram of Host Controller VI.vi, showing the temperature comparison logic that triggers the high-temperature alarm and invokes the Telegram alert Python script.*

5.5.2 Front Panel Dashboard

The front panel was designed as a complete monitoring dashboard, branded with the Department of Engineering and Al-Azhar University logos and titled **"Wireless Sensor Network (WSN) for Solar PV Thermal"**. It displays *Link Quality*, *Battery Voltage*, and *TC0*, alongside a graphical **Thermometer**, a **High Temp Alarm** round LED indicator, and a real-time **Waveform Chart** of temperature versus time.

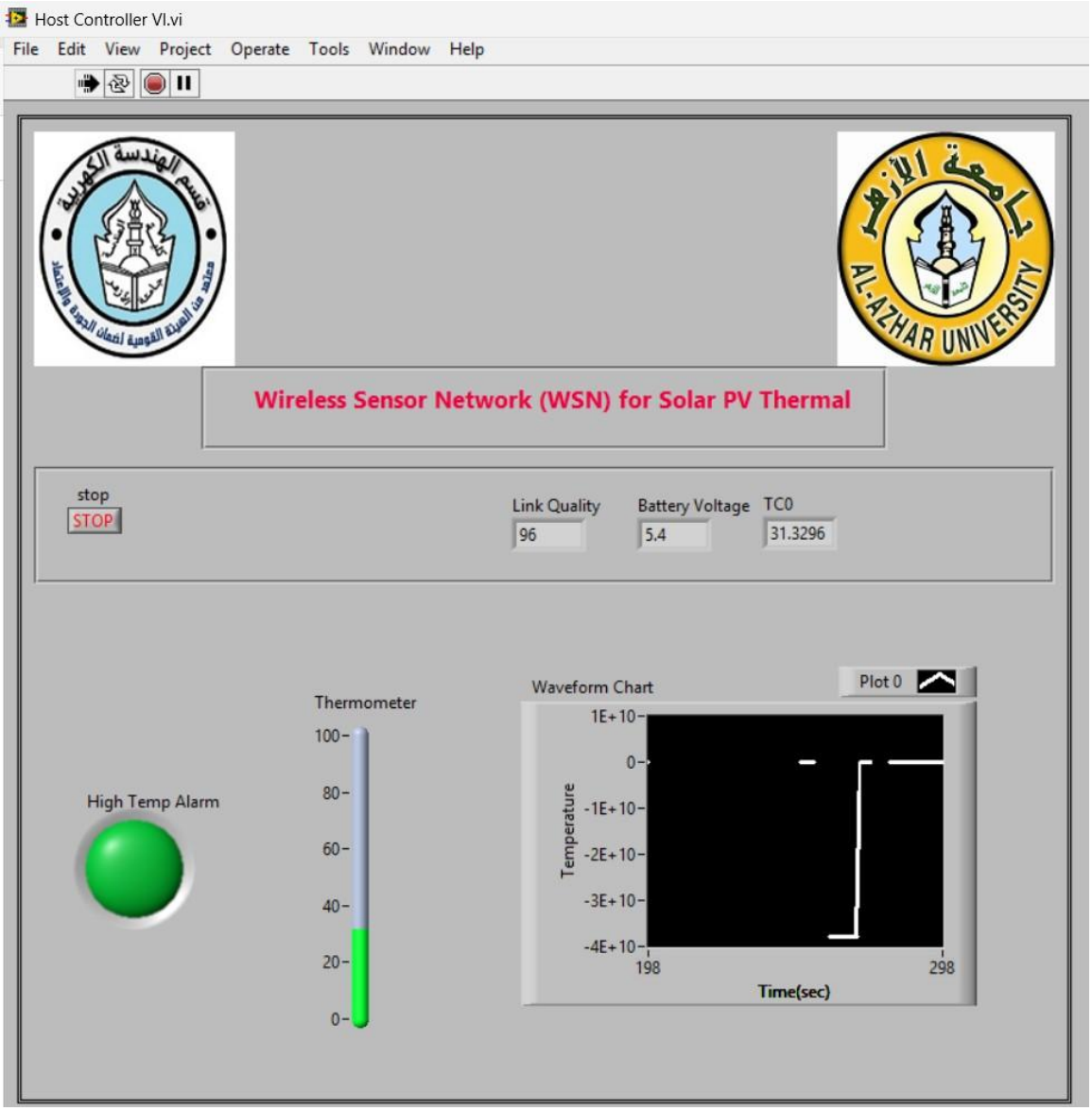

Figure 5.86: *Host Controller VI front panel during normal operation (TC0 = 31.33 °C, High Temp Alarm indicator OFF).*

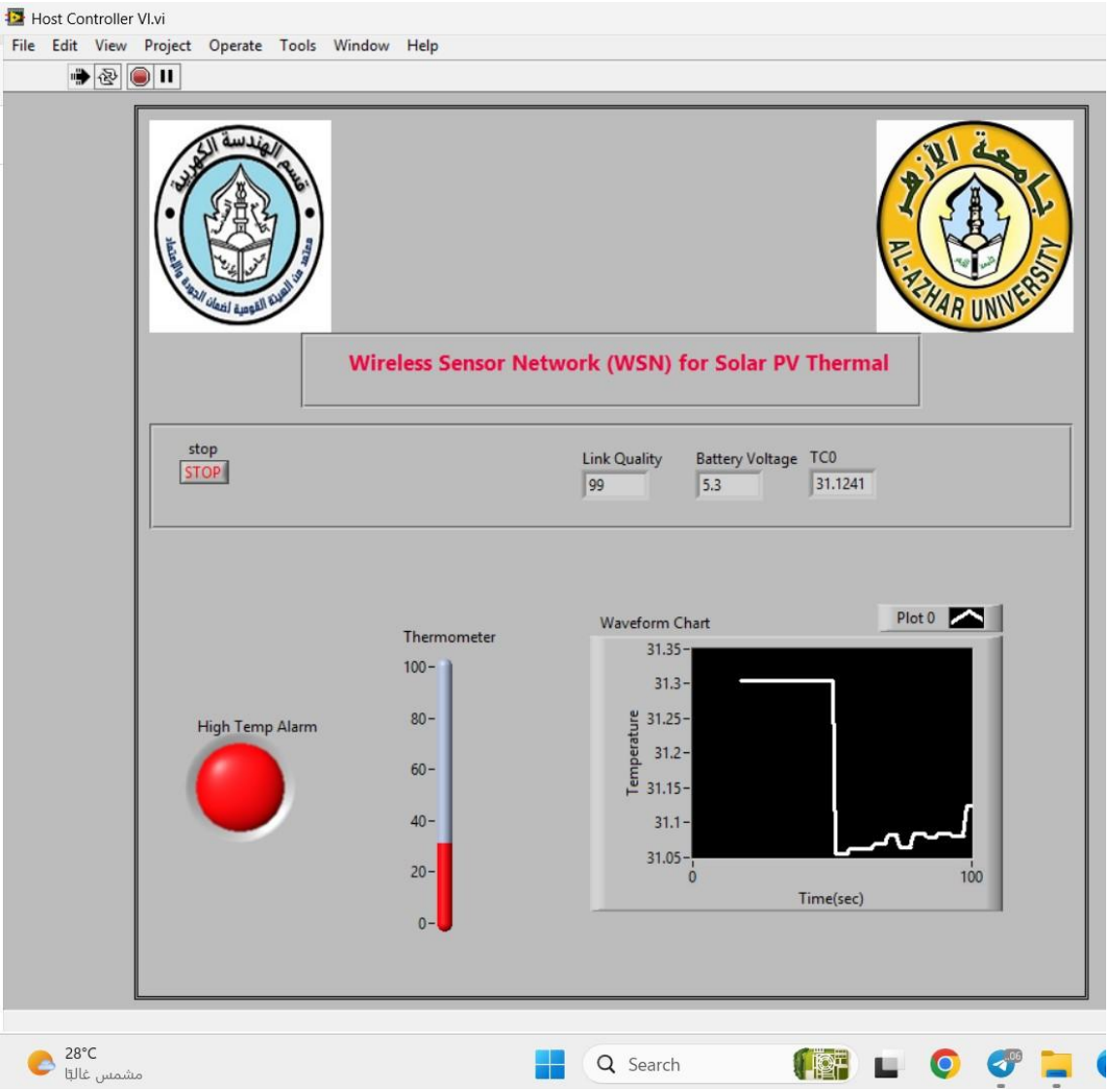

Figure 5.87: *Host Controller VI front panel with the High Temp Alarm indicator active, together with the live temperature trend on the waveform chart.*

5.5.3 Telegram Alert Script

When the high-temperature condition is detected, the Host Controller VI launches the Python script `telegram_alert.py`, passing the current TC0 reading as a command-line argument. The script compares the received temperature against a 35 °C threshold and, if exceeded, sends a formatted bilingual (English/Arabic) alert message to a Telegram group via the Telegram Bot API, in addition to sounding a local audible beep.

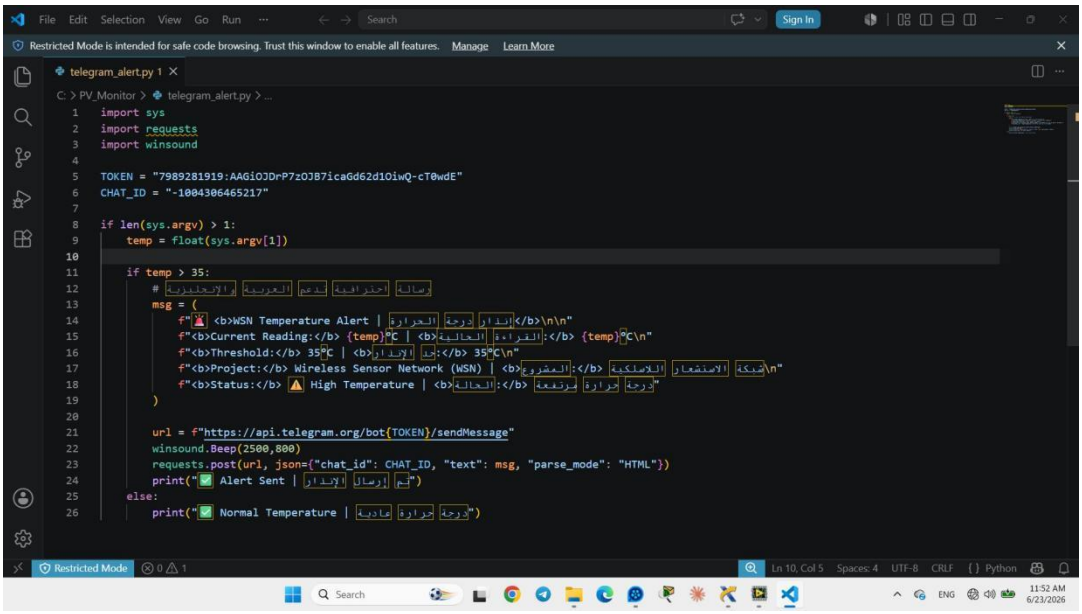
File Edit Selection View Go Run
Search
Sign In
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More
telegram_alert.py 1
C: > PV_Monitor > telegram_alert.py > ...
import sys
import requests
import winsound
TOKEN = "7989281919:AAGiOJDrP7zOJB7icaGd62d1OiwQ-cT0wdE"
CHAT_ID = "-1004306465217"
if len(sys.argv) > 1:
temp = float(sys.argv[1])
if temp > 35:
msg = (
f"WSN Temperature Alert | انذار درجة الحرارة\n\n"
f"Current Reading: {temp}°C | القراءة الحالية: {temp}°C\n"
f"Threshold: 35°C | حد الإنذار: 35°C\n"
f"Project: Wireless Sensor Network (WSN) | المشروع: الاستشعار اللاسلكية شبكة\n"
f"Status: ⚠ High Temperature | الحالة: درجة حرارة مرتفعة"
)
url = f"https://api.telegram.org/bot{TOKEN}/sendMessage"
winsound.Beep(2500,800)
requests.post(url, json={"chat_id": CHAT_ID, "text": msg, "parse_mode": "HTML"})
print("✅ Alert Sent | تم إرسال الإنذار")
else:
print("✅ Normal Temperature | درجة حرارة عادية")
Restricted Mode
Ln 10, Col 5 Spaces: 4 UTF-8 CRLF { } Python
Search
ENG
11:52 AM
6/23/2026

Figure 5.88: *Python `telegram_alert.py` script in Visual Studio Code — implementing Telegram Bot API notification with status-file-based duplicate suppression and an audible beep alert via `winsound`.*

5.5.4 Alert Verification

The complete alert pipeline was verified end-to-end by allowing the thermocouple reading to rise above the 35 °C threshold. As shown in Figure 5.89, the system successfully delivered a Telegram notification reporting a current reading of 54.76 °C, confirming that the LabVIEW host controller, the Python alert script, and the Telegram Bot API integration function correctly together.

Figure 5.89: *Telegram notification received on the WSN monitoring group, reporting a high-temperature reading of 54.76 °C against the 35 °C threshold.*

5.6 Summary

This chapter detailed the complete design and implementation process of the proposed WSN-based monitoring system, beginning with the creation of the LabVIEW project and the configuration of the NI-WSN gateway and nodes, followed by the development of monitoring VIs for single- and multi-node configurations. Experimental range testing demonstrated that the system maintains a usable wireless link up to a distance of approximately 50 m, with link quality degrading progressively with distance and the presence of physical obstructions such as walls. The system was further extended to support simultaneous thermocouple and soil-moisture sensing across two independent nodes, and was finally integrated with a Telegram-based alert mechanism capable of automatically notifying users of high-temperature events in real time, providing a practical and remotely monitorable wireless sensing solution.

Chapter 6

Conclusion and Future Work

Conclusion

This project presented the design and implementation of an Ad Hoc Wireless Sensor Network (WSN) using WSN 3212 nodes. The system was built to enable autonomous, infrastructure-free communication between sensor nodes, allowing real-time data collection and monitoring without reliance on a fixed network backbone.

The WSN 3212 nodes were configured to operate in a mesh topology, where each node functions simultaneously as a data source and a relay point. This multi-hop communication architecture ensures network resilience, as the failure of a single node does not disrupt the overall system operation. The nodes were programmed to periodically collect environmental data and transmit it through the network gateway.

The gateway served as the central point of data aggregation, receiving readings from all active nodes and forwarding the processed data to the monitoring system. LabVIEW was utilized as the primary interface for real-time visualization, providing graphical dashboards that display network status, node connectivity, and sensor readings in a clear and user-friendly manner.

To enhance the system's responsiveness, a Python-based alert pipeline was developed and integrated with the Telegram messaging platform. This pipeline continuously monitors incoming sensor data and automatically triggers instant notifications to the project team whenever abnormal readings or node failures are detected, ensuring timely intervention without the need for constant manual monitoring.

The overall system architecture demonstrated strong performance in terms of reliability, low latency, and ease of deployment. The Ad Hoc nature of the network eliminates the dependency on pre-existing infrastructure, making it highly suitable for dynamic environments such as disaster recovery zones, remote field monitoring, and temporary industrial deployments.

The results confirmed that the proposed WSN-based Ad Hoc network successfully meets the project objectives, providing a scalable, cost-effective, and intelligent monitoring solution that combines wireless sensor technology, real-time data visualization, and automated alerting into a unified system.

Future Work

Despite the promising results achieved, several directions can be explored to further enhance the system:

- **Scalability Testing:** Expanding the network to include a larger number of WSN nodes to evaluate performance under higher traffic loads and longer communication ranges.
- **Energy Harvesting Integration:** Incorporating solar panels or energy harvesting modules to power the WSN nodes, enabling fully autonomous and self-sustaining operation.
- **Advanced Routing Protocols:** Implementing and comparing dynamic routing protocols such as AODV or DSDV to optimize data delivery and reduce latency in the Ad Hoc network.
- **Security Enhancements:** Adding encryption and authentication mechanisms to secure data transmission between nodes and protect against unauthorized access.
- **Cloud Integration:** Connecting the gateway to a cloud platform for remote data storage, long-term analysis, and visualization through web-based dashboards.
- **Mobile Node Support:** Extending the system to support mobile sensor nodes, enabling dynamic topology changes and improving adaptability in real-world deployment scenarios.

References

Bibliography

[1] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, “Wireless Sensor Networks: A Survey,” *Computer Networks*, vol. 38, no. 4, pp. 393–422, 2002.

[2] C.E. Perkins, *Ad Hoc Networking*. Addison-Wesley, 2001.

[3] Aginode, “WSN 3212 Wireless Sensor Node – Technical Datasheet,” Aginode GmbH, 2015.

[4] National Instruments, “LabVIEW User Manual,” National Instruments Corporation, Austin, TX, 2020. [Online]. Available: `https://www.ni.com/en/support/documentation.html` [Accessed: June 2026]

[5] python-telegram-bot Contributors, “python-telegram-bot Documentation,” [Online]. Available: `https://python-telegram-bot.readthedocs.io` [Accessed: June 2026]

[6] C.E. Perkins and E.M. Royer, “Ad Hoc On-Demand Distance Vector Routing,” in *Proc. 2nd IEEE Workshop on Mobile Computing Systems and Applications*, pp. 90–100, 1999.

[7] IEEE Standard, “IEEE 802.15.4: Standard for Low-Rate Wireless Networks,” IEEE, New York, 2020. [Online]. Available: `https://standards.ieee.org/ieee/802.15.4` [Accessed: June 2026]

[8] K. Zimmermann, “Understanding Mesh Network Topologies in WSN,” *IoT Analytics Journal*, vol. 5, no. 2, pp. 14–27, 2023.

[9] S. Sudevalayam and P. Kulkarni, “Energy Harvesting Sensor Nodes: Survey and Implications,” *IEEE Communications Surveys & Tutorials*, vol. 13, no. 3, pp. 443–461, 2011.

[10] Amazon Web Services, “AWS IoT Core Documentation,” [Online]. Available: https://docs.aws.amazon.com/iot [Accessed: June 2026]

[11] Y. Wang, G. Attebury, and B. Ramamurthy, “A Survey of Security Issues in Wireless Sensor Networks,” *IEEE Communications Surveys & Tutorials*, vol. 8, no. 2, pp. 2–23, 2006.

[12] W.R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, “Energy-Efficient Communication Protocol for Wireless Microsensor Networks,” in *Proc. 33rd Hawaii International Conference on System Sciences*, 2000.

[13] ZigBee Alliance, “ZigBee Specification,” ZigBee Alliance, 2015. [Online]. Available: https://zigbeealliance.org/solution/zigbee [Accessed: June 2026]

[14] C. Intanagonwiwat, R. Govindan, and D. Estrin, “Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks,” in *Proc. ACM MobiCom*, pp. 56–67, 2000.

[15] J. Yick, B. Mukherjee, and D. Ghosal, “Wireless Sensor Network Survey,” *Computer Networks*, vol. 52, no. 12, pp. 2292–2330, 2008.

[16] H. Liu, A. Nayak, and I. Stojmenovic, “Sensor Deployment and Node Localization Models in Wireless Sensor Networks,” *IEEE Transactions on Parallel and Distributed Systems*, 2003.

[17] National Instruments, “Real-Time Monitoring with LabVIEW,” [Online]. Available: https://www.ni.com/en/solutions/real-time-monitoring.html [Accessed: June 2026]

[18] Python Software Foundation, “Python 3 Documentation,” [Online]. Available: https://docs.python.org/3 [Accessed: June 2026]

[19] Telegram, “Telegram Bot API Documentation,” [Online]. Available: https://core.telegram.org/bots/api [Accessed: June 2026]

[20] J.F. Kurose and K.W. Ross, *Computer Networking: A Top-Down Approach*, 7th ed. Pearson, 2017.

[21] T.S. Rappaport, *Wireless Communications: Principles and Practice*, 2nd ed. Prentice Hall, 2002.

[22] L. Atzori, A. Iera, and G. Morabito, “The Internet of Things: A Survey,” *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[23] W. Ye, J. Heidemann, and D. Estrin, “An Energy-Efficient MAC Protocol for Wireless Sensor Networks,” in *Proc. IEEE INFOCOM*, vol. 3, pp. 1567–1576, 2002.

[24] NS-3 Consortium, “NS-3 Network Simulator Documentation,” [Online]. Available: `https://www.nsnam.org/documentation` [Accessed: June 2026]

[25] F. Koushanfar, M. Potkonjak, and A. Sangiovanni-Vincentelli, “Fault Tolerance Techniques for Wireless Ad Hoc Sensor Networks,” in *Proc. IEEE Sensors Conference*, 2002.

تمهيد

أدى النمو المستمر لأنظمة الطاقة المتجددة، ولا سيما أنظمة الطاقة الشمسية الكهروضوئية/الحرارية الهجينة (PV/T)، إلى ظهور حاجة ملحة لتوفير حلول مراقبة ذكية ومنخفضة التكلفة وموثوقة، قادرة على تتبع السلوك الكهربائي والحراري لهذه الأنظمة في الوقت الفعلي. وتُعد الألواح الشمسية شديدة التأثر بالظروف البيئية المحيطة، إذ قد يتراجع أداؤها بشكل ملحوظ دون أن يكون المشغل على علم بحدوث أي مشكلة. وانطلاقاً من هذه الحاجة، تم تنفيذ مشروع التخرج هذا بعنوان «**بناء شبكة مخصصة** (Ad Hoc) (Network) باستخدام شبكات الاستشعار اللاسلكية»، بهدف تصميم وتنفيذ نظام مراقبة لاسلكي قادر على الرصد المستمر للمتغيرات الكهربائية والحرارية الأساسية لمنظومة الطاقة الشمسية الكهروضوئية/الحرارية (PV/T)، وإبلاغ المستخدم بحالة النظام بصورة آنية وفي الوقت الحقيقي.

يوثق هذا التقرير المسيرة الكاملة للمشروع، بدءًا من الأسس النظرية لشبكات الحاسوب وشبكات الاستشعار اللاسلكية، مروراً بالصياغة المنهجية لمشكلة المراقبة، وصولاً إلى تصميم وتنفيذ نظام مراقبة متكامل وفعّال قائم على شبكات الاستشعار اللاسلكية (WSN). وقد تم بناء النظام المنفذ اعتمادًا على منصة شبكات الاستشعار اللاسلكية التابعة لشركة (National Instruments (NI)، والتي تتكون من بوابة اتصال رئيسية من نوع NI WSN-9791 Gateway وعقدتي استشعار لاسلكيتين من الطرازين NI NI WSN-3212 WSN-3202 ، وتعمل هذه المكونات معًا على قياس درجة الحرارة، وشدة الإشعاع الشمسي، بالإضافة إلى الجهد والتيار لكل من المصفوفة الشمسية والحمل الكهربائي. ويتم بعد ذلك جمع البيانات ومعالجتها وعرضها باستخدام برنامج LabVIEW ، بينما تُرسل القراءات الحرجة والتنبيهات إلى المستخدم عبر قناة تعتمد على تطبيق Telegram ، مما يتيح متابعة حالة النظام عن بُعد وبصورة آنية من أي مكان.

تم تنظيم هذا التقرير في خمسة فصول. يقدم **الفصل الأول** المفاهيم الأساسية لشبكات الحاسوب، بما في ذلك مكونات الشبكات، وطوبولوجيات الشبكات، والأنواع المختلفة للشبكات التي تشكل الإطار العام الذي تعمل ضمنه شبكات الاستشعار اللاسلكية. أما **الفصل الثاني** فيركز على شبكات الاستشعار اللاسلكية بشكل خاص، موضحًا مفهومها، وبنيتها المعمارية، وطبقات البروتوكولات الخاصة بها، إضافة إلى المزايا والتحديات المرتبطة بتصميمها وتطبيقها.

ويستعرض **الفصل الثالث** برنامج LabVIEW بوصفه المنصة البرمجية المستخدمة لجمع البيانات ومعالجتها، موضحاً دوره في الربط بين مكونات العتاد الخاصة بشبكة الاستشعار اللاسلكية وعرض البيانات للمستخدم. أما **الفصل الرابع** فيقدم الصياغة الرسمية للمشكلة التي استند إليها تصميم النظام، من خلال تحويل متطلبات مراقبة منظومة الطاقة الشمسية الكهروضوئية/الحرارية (PV/T) إلى نموذج رياضي متكامل يتضمن دوال الهدف، وقيود التصميم، ودراسة حالة عددية مستندة إلى العتاد الفعلي المستخدم في المشروع.

وأخيرًا، يوثق **الفصل الخامس** التصميم والتنفيذ الكاملين للنظام، بما في ذلك بنية عقد الاستشعار اللاسلكية، وطوبولوجيا الشبكة، وإعدادات العتاد التي تم تنفيذها باستخدام برنامج NI Measurement & Automation Explorer (NI MAX)» بالإضافة إلى التطبيق البرمجي باستخدام LabVIEW وقناة المراقبة المعتمدة على Telegram» والمستخدمة لإرسال التنبيهات الفورية.

ونأمل أن لا يقتصر هذا التقرير على توثيق العمل التقني المنجز خلال هذا المشروع فحسب، بل أن يكون أيضا مرجعًا واضحًا ومفيدًا لكل من يرغب في فهم الكيفية العملية لتطبيق شبكات الاستشعار اللاسلكية في مراقبة أنظمة الطاقة المتجددة، والاستفادة منها في تطوير حلول مراقبة ذكية وفعّالة لهذا المجال الحيوي.

جامعة الأزهر الشريف

كلية الهندسة

قسم الهندسة الكهربية

(اتصالات و إلكترونيات)

مشروع التخرج للدرجة العلمية
البكالوريوس

بناء شبكة مؤقتة ذاتية التنظيم باستخدام شبكات الاستشعار اللاسلكية

تحت إشراف :

أ.د.م / عبداللطيف إبراهيم مصطفى

مقدم من :

محمد سعيد السيد خالد (قائد المجموعة)

أحمد السيد رمضان عبدالحميد

محمد طه أحمد أبوالهدى طه

أنس مصطفى عبدالحميد غازي

محمد عبدالفتاح محمد محمد بحيري

إسلام أحمد الصافي أحمد

يوليو ٢٠٢٦